Calhoun: The NPS Institutional Archive

Theses and Dissertations | Thesis Collection

2011-03

Automatically detecting authors' native language

Ahn, Charles S.

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/5821

http://www.nps.edu/library

Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

AUTOMATICALLY DETECTING AUTHORS' NATIVE LANGUAGE

by

Charles S. Ahn

March 2011

Thesis Advisor:	Craig H. Martell
Second Reader:	Pranav Anand

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704–0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
7–3–2011	Master's Thesis	2009-03-01—-2011-03-25

4. TITLE AND SUBTITLE

Automatically Detecting Authors' Native Language

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Charles S. Ahn

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of the Navy

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. **IRB Protocol Number: N/A**

14. ABSTRACT

When non-native speakers learn English, their first language influences how they learn. This is known as L1-L2 language transfer, and linguistic studies have shown that these language transfers can affect writing as well. If there were a model that exploits L1-L2 language transfer to identify the authors' native language, it would be an invaluable tool for the intelligence community as well as in the field of education. Therefore, the objective of this research is to find out if it is possible to automatically detect the author's native language based on his/her writing in English using traditional machine learning techniques. For this research, we used eight different collections of writings by speakers of eight different nationalities: native English speakers as well as speakers of Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish. Among the various feature sets used in this research, character trigrams and bag of words alone achieved higher than 80% accuracy, and the empirical analysis of character trigrams revealed that the character trigrams just model lexical usage. When content words were extracted, the performance dropped and the results revealed that the topic words were doing all the work.

15. SUBJECT TERMS

Machine Learning, Natural Language Processing, Supervised Learning, Naive Bayes, Maximum Entropy, L1-L2 Language Transfer, Native Language Detection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	UU	115	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

AUTOMATICALLY DETECTING AUTHORS' NATIVE LANGUAGE

Charles S. Ahn
Lieutenant, United States Navy
B.S., Binghamton University, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2011

Author: Charles S. Ahn

Approved by: Craig H. Martell
Thesis Advisor

Pranav Anand
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

When non-native speakers learn English, their first language influences how they learn. This is known as L1-L2 language transfer, and linguistic studies have shown that these language transfers can affect writing as well. If there were a model that exploits L1-L2 language transfer to identify the authors' native language, it would be an invaluable tool for the intelligence community as well as in the field of education. Therefore, the objective of this research is to find out if it is possible to automatically detect the author's native language based on his/her writing in English using traditional machine learning techniques. For this research, we used eight different collections of writings by speakers of eight different nationalities: native English speakers as well as speakers of Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish. Among the various feature sets used in this research, character trigrams and bag of words alone achieved higher than 80% accuracy, and the empirical analysis of character trigrams revealed that the character trigrams just model lexical usage. When content words were extracted, the performance dropped and the results revealed that the topic words were doing all the work.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

First and foremost, I would like to thank my advisor, Professor Craig Martell, for giving me the opportunity to pursue incredibly interesting work, providing all the resources I needed, and giving me the encouragement and support necessary in an undertaking of this magnitude. I would also like to thank Professor Pranav Anand for all of his support and guidance. His valuable insight and recommendations helped to make this thesis ”outstanding.” I would like to thank Dr. Andrew Schein for his technical support, and Dr. Na Rae Han for providing invaluable resources that helped me to start this thesis in the right direction. Finally, I would like to thank Ms. Nina Liakos for editing my thesis.

I would like to thank my NLP lab partners, CDR Grady and LT Boutwell, with who I have shared the lab room for over nine months. Their humors and stories about the Navy made the hours in the lab pass much more enjoyable.

Lastly, I would like to thank my wife, Young Ji Kim. Without your support and understanding, this work would never have been completed.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

1.1 Motivation

Writing in English has always been a difficult task for non-native speakers. Even after studying the mechanics and grammar rules for years, it is a very difficult task for non-native speakers to write with the same natural flow found in writings by native speakers. Moreover, when non-native speakers write, they leave trails, whether mistakes or just a unique pattern influenced by their first language, which is known as first language (L1) - second language (L2) language transfer. For example, in Korean, there is no concept of using articles like *"a"* or *"the"*, so it is very likely that native Koreans will misuse or misplace articles in their English writing. If there are enough data that represent how native Koreans write in English, we can build a model, using language processing techniques, that captures how native Koreans write by focusing on the unique patterns that distinguish them from other people who speak different languages. If we can build these models for all languages, in theory, we will be able to identify authors' native languages just based on their writing style in English.

Although, as far as we know, a type of system that detects an author's native language based on their writing has not been a critical application in any field, but as the world becomes more connected than ever, and as sharing information is gets continuously easier, being able to discover the native language of the author of a threatening message could be a significant tool for capturing the people who are responsible for such the threat. For example, if a message describing a possible terrorist activity is intercepted, the FBI and the intelligence community can use automatic language detecting capability to learn more about the threat, such as who may be behind it.

The education field uses these language models in various ways. The ETS corporation has been researching language models that can help them to build their own system for automatically evaluating essays for TOEFL exams that are tailored to the student's native language (Na Rae Han, p.c.). Also, these language models can help ESL teachers to tailor their teaching methods to the students' native language. For example, Korean native speakers and Spanish native speakers will likely have different patterns of writing and different kinds of problems, and if these language models provides this information to ESL teachers, they can help the students

more effectively.

Detecting the authors' native language is relatively a new topic in the natural language processing field. A little research has been done in this domain, but it is still far from accomplishing the tasks described above. In this research, we wish to answer three questions: 1. Given essays written by non-native speakers, how well can we detect the authors' native languages using various natural language processing tools? 2. What is the strongest feature set and why does this particular feature set work better than the other feature sets? 3. To what extend is the second question dependent on the topics discussed in the corpus? This is a very important question because if all Chinese essays were about technology, then the problem would just be detecting what they wrote instead of how they write.

1.2 Organization

We have organized this thesis as follows: In Chapter 1, we provide the motivation for this research. In Chapter 2, we provide 1) an overview of L1-L2 language transfer at the lexical (vocabulary) and syntactic (sentence structure) levels, 2) an overview of feature sets, 3) general natural language processing techniques as well as evaluation methods, and 4) prior related works. In Chapter 3, we detail our technical approach, including a discussion of the corpora used, the feature sets used, and the set-up of our experiments using this data along with the classification methods. In Chapter 4, we present the results of our experiments as well as a discussion of their significance. We begin by with discussing the results of the various feature sets and comparing the performances of the maximum entropy and Naive Bayes classifications. We then analyze character trigrams and study what drives their success by empirical analysis, followed by a review of the performances of a lexical feature and character n-grams and their relationships. Lastly, we discuss the role of topics in discriminating between the authors and how the results change when topics are controlled. In Chapter 5, we conclude with a summary of our work along with recommendations for future research.

1.3 Results

In this research, we used two corpora, *International Corpus of Learner English (ICLE)* and *Centre for English Corpus Linguistics (CECL)*, written by speakers of eight different nationalities: native English speakers as well as speakers of Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish to identify writers L1 [17]. Overall, we achieved higher than 80% accuracy using either character trigrams or bag of words alone as a feature set when Maximum

Entropy was used as the machine learning technique. Syntactical feature sets such as POS n-grams and distribution of transformation rules worked fairly well for detecting Chinese and Japanese, but it performed less well with Slavic and Romance languages. Empirical analysis of character trigrams also demonstrated that character trigrams only model lexical usage, leading us to conclude that the best indication for detecting authors' native languages is their lexical usage. Furthermore, to find out to what extent lexical usage is dependent on the topics discussed in the corpus, we used the LDA model to show that the distribution of topics of each language corpus is distinct from other distributions, which indicated that the topics are actually doing most of the work. Then we used TF-IDF techniques to identify and extract the top content words, and as the content words are extracted, the performance of the lexical model and the character n-grams dropped with respect to the size of words extracted. In other words, as the topics were extracted, the performance dropped; this phenomenon supported our hypothesis that the topics were doing the work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Prior and Related Work

2.1 Introduction

Learning to automatically determine a non-native speaker's native language (L1) based on his or her writing in English requires us to understand the phenomenon of how different native languages uniquely affect learning a second language (L2). Therefore, in this chapter, we present concepts of L1-L2 language transfer that are relevant to detecting an author's L1 using machine learning techniques. Once this foundation is discussed, we define the feature sets that are used in this research, followed by an overview of machine learning classification techniques. Then, we discuss different types of evaluation methods, tools, and information retrieval techniques. The chapter concludes by surveying the prior and related works that have been published.

2.2 Language Transfer

When non-native speakers learn English as their second language, in general, it is very difficult for the learners to become fluent in English in both writing and speaking. Linguists have come up with several different explanations as to why L2 acquisition is difficult and what influences learning L2. One of the major influences in L2 acquisition is a learner's L1. Each language has a unique structure, and there is evidence that a learner's L1 interferes with learning L2, which is known as L1-L2 language transfer. Terence Odlin discusses the cross-linguistic influences in language learning in reference [1], and some of his discussions that are related to this thesis will be discussed in the next few sections.

2.2.1 Lexical Transfer

Odlin says that learners that have a large lexicon in common between L1 and L2 will adapt to the L2 faster than learners with an L1 that does not share a large common lexicon with the L2, and this phenomenon is known as *lexical transfer*. For example, the word *justify* can be written as *justifier* in French, so French speakers will have an easier time learning what justify means in English than Koreans, whose language has few lexical similarities with English. Lexical transfer also contains morphological and syntactic information [1]. An example of morphological transfer is the similar English and Spanish suffixes *-ous* and *-oso* in words *scandalous* and

escandaloso. The similar suffixes will help Spanish speakers to identify cognates. Syntactic transfer is discussed in the next section.

2.2.2 Syntax

Word Order

Most human languages have one of the following basic word orders: Verb-Subject-Object (VSO), Subject-Verb-Object (SVO), or Subject-Object-Verb (SOV). Although the idea of whether the L1's basic word order influences the learning of an L2 is arguable, Odlin argues that if the learner's L1 word order is different from that of the L2, it will be likely to affect the L2 acquisition [1]. For example, Philippine speakers of languages such as Ilocano and Tagalog, which are SOV, showed patterns of SOV word orders in their English writing. Also, native Japanese showed SOV patterns in their English writing, which is consistent to Japanese word order. Odlin also says that the word order within the clause may influence the acquisition of the L2. In English noun phrases (NP), articles and modifiers precede nouns (e.g., the beautiful house). However, other languages have their own rules governing the positions of adjectives, adverbs, and other word classes, and there is evidence that different placement of modifiers also influences L2 acquisition [1]. For example, a survey of Hebrew speakers found that there is a strong tendency for speakers to misplace adverbial elements, which follows the Hebrew writing pattern, as seen in the following error: *I like very much movies.*

Relative Clauses

Some language structures place relative clauses on the right side of the head noun, which is known as the Right Branching Direction (RBD); on the other hand, other language structures place relative clauses on the left of the head noun, which is known as the Left Branching Direction (LBD). English is an example of a language that relies on RBD, and Japanese is an example of LBD. Odlin used the example in Figure 2.1 to explain the difference between English and Japanese in terms of placing relative clauses.

The cheese that the rat ate was rotten
Nezumi ga ttabeta cheese wa kusatte ita
rat ate cheese rotten was

Figure 2.1: Right Branching Direction vs Left Branching Direction

In Figure 2.1, the head noun is *cheese* and the relative clause *that the rat ate*, which modifies

the head noun, is placed to the right of *cheese*; however, in Japanese, *rat ate*, which modifies *cheese*, is placed before *cheese*. Odlin argues that if the L2 uses a different branching direction than that of the learner's L1, it becomes more difficult for that learner to adopt L2 than it is for those learners whose L1 does not have that difference. Spanish uses RBD just as English does, and the fact that Spanish learners of English have greater success repeating such sentences than Japanese native speakers supports Odlin's argument of how branch direction affects the L2 acquisition.

2.3 Features

We have discussed how learners' L1 can influence their L2 acquisition because this concept can be used to predict non-native speakers' native language based on how they write in English. However, how do we know which characteristics are most useful in predicting the L1? Choosing the right set of characteristics (or "features," as they are called in machine learning) is very important, and this section presents a variety of useful features that are used in this research.

2.3.1 Lexical Features

Lexical features are the most straightforward features that simply exploit authors' choice of the words they used. There are many different types of lexical features, but in this research, we discuss just two that are most relevant to this research.

Bag of Words

Among the lexical features, the "bag of words" is the most straightforward, since it simply measures the frequency of each word regardless of how words are ordered. The bag of words has been widely used in natural language processing problems such as authorship attribution because it is simple and also captures authors' preferences in terms of word usage. If a particular author tends to use a particular word that is unique to that author, then the bag of words captures that. For the same reason, a distribution of word frequency from a collection of documents written by Chinese writers can be very different from the distribution of word frequency from documents written by Bulgarian writers.

Function Words

Words can be divided into two classes: *function words* and *content words*. Content words are words such as nouns, verbs, adjectives, and most adverbs [2]. Content words are subject to change over time, and the choice of content words is heavily dependent on semantics. In contrast, function words are words such as articles, prepositions, pronouns, numbers, conjunctions,

auxiliary verbs, and certain irregular forms. They have specific syntactic functions governed by grammatical rules, and they are used to construct grammatical sentences out of individual words. Also, some function words, such as articles and prepositions, carry important semantic information, as do past and future tenses. Efstathios Stamatatos stated in [3] that the features from function words are highly discriminative in the authorship attribution problems.

2.3.2 Syntactic Features

Syntactic features are used in authorship attribution problems because they capture authors' unique syntactic patterns. Stamatatos states in [3] that authors tend to use similar syntactic patterns unconsciously. Therefore, it is logical to consider syntactic features in the scope of this research, and we discuss two different types of syntactic features that are relevant to this research.

Distribution of Transformation Rules

Word order is one component of the syntactic structure of a sentence, or the rules by which word combinations form acceptable sentences. Acceptable patterns of word combination can be given by a tree structure, which specifies the relationships between words and phrases, as shown in Figure 2.2. There is much debate in formal linguistics about the proper tree structures for sentences, as well as much research in computational linguistic about how to generate parses efficiently. In this work, we assume that constituent trees (Chomsky 1957) are in an appropriate representation for syntactic features, and extract such representations using the Stanford Parser. Once a sentence is parsed, syntactic rules, also known as transformation rules, are extracted from the parsed tree. Then using the distribution of these transformation rules as a feature set may capture an unique syntactic patterns from a group that is particular do that group.

Learning language is difficult

```
(ROOT
  (S
    (NP
      (NP (NNP Learning))
      (NP (DT a) (JJ new) (NN language)))
    (VP (VBZ is)
      (ADJP (JJ difficult)))
    (. !)))
```

Figure 2.2: Stanford parser output

For example, using the parsed tree in Figure 2.2, the following transformation rules are extracted: $S \rightarrow NPVP$, $NP \rightarrow NPNP$, and $VP \rightarrow ADJP$. The first expression states that a sentence (S) is constituted by a noun phrase (NP) followed by a verb phrase (VP), and the second rule states that a noun phrase is constituted by a noun phrase followed by another noun phrase. These transformation rules describe both what the syntactic class of each word is and how the words are combined to form phrases or other structures.

Part-of-Speech (POS) N-Grams

Words in a sentence can be broken down into classes based on their syntactic and morphological functions. These classes are known as parts of speech. The Penn Treebank tagset contains 36 POS tags and 12 other tags as shown in Table 2.1 [4]. The list of the Penn Tree POS tagset is presented, since this tagset is used by the Stanford parser, which is the tool used for POS tagging in this research.

1.	CC	Coordinating conjunction	25.	TO	to
2.	CD	Cardinal number	26.	UH	Interjection
3.	DT	Determiner	27.	VB	Verb, base form
4.	EX	Existential there	28.	VBD	Verb, past tense
5.	FW	Foreign word	29.	VBG	Verb, gerund / present participle
6.	IN	Preposition / subordinating conjunction	30.	VBN	Verb, past participle
7.	JJ	Adjective	31.	VBP	Verb, non-3rd ps. sing. present
8.	JJR	Adjective, comparative	32.	VBZ	Verb, 3rd ps. sing. present
9.	JJS	Adjective, superlative	33.	WDT	wh-determiner
10.	LS	List item marker	34.	WP	wh-pronoun
11.	MD	Modal	35.	WP$	Possessive wh-pronoun
12.	NN	Noun, singular or mass	36.	WRB	wh-adverb
13.	NNS	Noun, plural	37.	#	Pound sign
14.	NNP	Proper noun, singular	38.	$	Dollar sign
15.	NNPS	Proper noun, plural	39.	.	Sentence-final punctuation
16.	PDT	Predeterminer	40.	,	Comma
17.	POS	Possessive ending	41.	:	Colon, semi-colon
18.	PRP	Personal pronoun	42.	(	Left bracket character
19.	PP$	Possessive pronoun	43.	)	Right bracket character
20.	RB	Adverb	44.	”	Straight double quot
21.	RBR	Adverb, comparative	45.	’	Left open single quote
22.	RBS	Adverb, superlative	46.	“	Left open double quote
23.	RP	Particle	47.	’	Right close single quote
24.	SYM	Symbol (mathematical or scientific)	48.	”	Right close double quote

Table 2.1: The Penn Treebank POS tagset

As we have seen above, Figure 2.2 also provides POS tags, which are located right next to the words being tagged. For instance, *Learning* is tagged with *NNP* and the article *a* is tagged with *DT*. These POS tags are extracted sequentially from the parsed trees to use POS n-grams as feature sets in this research. A POS unigram is comprised of a single POS; a POS bigram is a pairing of two adjacent POSs, and a POS trigram is three consecutive POSs. POS n-grams have been used in other natural language processing (NLP) researches because they provide a hint of the structural analysis of a sentence. For example, a typical standard English sentence will generate high counts of POS tags for determiners follow by tags for nouns, but the same sequence of POS tags may not appear as much from writings by Japanese since there is no concept of articles in the Japanese language.

2.3.3 Character N-Grams

Character unigrams, bigrams and trigrams are just like POS unigrams, bigrams and trigrams but with individual characters instead of POSs. Character N-grams have been widely used as a feature set in many natural language processing studies because they can capture nuances of style including lexical information, hints of contextual information, use of punctuation and capitalization [3]. Additionally, such n-grams are noise tolerant. That is, when texts contain grammatical errors or non-standard use of punctuation, the character n-gram is not affected. For example, the words *hello* and *helo* would generate many common character trigrams, but in a lexical-based representation, they would just be two different types. Character n-grams also capture errors that could be used to discriminate between the different data groups. Large n-grams are better at capturing lexical and contextual information, but the larger n-grams substantially increase the dimensionality. On the other hand, small (2 or 3) n-grams could capture sub-word information but would not be adequate for representing the contextual information [3].

2.4 Machine Learning Tools

Once a feature set is selected for inputs, the next logical step is to choose a machine learning technique that will process these inputs. Although there are many different machine learning techniques, this paper focuses on only two methods that are used in this research: *Naive Bayes* and *Maximum Entropy*. The following sections will provide an overview of each technique.

2.4.1 Naive Bayes

Naive Bayes is a simple probabilistic classifier based on Bayes' Theorem with a strong independence assumption that works in a supervised learning setting [5]. Bayes' rule, Equation 2.1, is used to predict the likelihood of a class *C*, given features *F*.

$$p(C|F_1, ..., F_n) = \frac{p(C)p(F_1, ..., F_n|C)}{p(F_1, ..., F_n)} \tag{2.1}$$

Since the numerator can be written as joint probability, as the number of features gets bigger, the application of this method gets very expensive. This is where independence assumptions come into play. Naive Bayes says that all features are independent of each other; therefore, the numerator of Equation 2.1 can be re-written as Equation 2.2. Also, since *p(F)* in equation 2.1 is constant and does not affect the likelihood, Equation 2.2 omits the denominator in Equation 2.1.

$$p(C)\prod_{i=1}^{n} p(F_i|C) \tag{2.2}$$

$$c^* = argmax P(c)\prod_{i=1}^{n} p(f_i|c) \tag{2.3}$$

The most probable class, c*, is returned by the function *argmax*, which returns the value from the x-axis where the respective values from the y-axis are highest.

Smoothing

A probabilistic classifier such as Naive Bayes works very well, if the models are trained by the complete data that represents the subject being classified; however, models are trained by the particular data set and use those available data to compute the maximum likelihood estimation (MLE), which is the most probable value based on the data available. Therefore, there is good chance that the test data may have data that never appeared in the training data, which would result zero probability. In order to avoid the zero-probability issue, smoothing techniques are used. There are many smoothing algorithms, but only two smoothing techniques will be covered in this section.

Laplace Smoothing

Laplace smoothing is the easiest smoothing technique to implement, but it does not work well enough to be widely used in modern models. The unsmoothed version of maximum likelihood

estimate of probability is normalized by dividing the seen feature counts, C_i, by the total number of tokens, *N*, as shown in Equation 2.4 [5].

$$P(w_i) = C_i/N \tag{2.4}$$

The probability of unseen features would be zero, so Laplace smoothing adds one to each count and increases the denominator by the total count of type *T* as follows:

$$P_{Laplace}(W_i) = \frac{C_i + 1}{N + T} \tag{2.5}$$

A problem with Laplace Smoothing is that it gives too much probability mass to unseen feature counts, and consequently it is known to perform less well than the smoothing techniques in practice; therefore, we introduce another smoothing technique called Good-Turing Smoothing in the next section, which improves the Laplace smoothing technique's shortfall, and the Good-Turning smoothing technique is used in this research.

Good-Turing Smoothing

Instead of adding 1 to each count as Laplace Smoothing, the Good-Turing algorithm is based on computing c and N_c, where c is a count of occurrences. For example, if the word *the* only shows once in a set of data, the value for c will be 1, and we will use the term *frequency c* to refer to the value of c. N_c is the number of counts that occur c times. If there are 5 different features that are seen only once (c = 1), N_1 would be 5. N_c is also known as the *frequency of frequency c* [5].

$$N_c = \sum_{x.count(x)=c} 1 \tag{2.6}$$

Good-Turing smoothing uses an intuition that the probability of a feature that occurred c times in the training data can be estimated by using a count that occurred c+1 times. The probability of unseen counts can be estimated by the probability of being seen just once. Using this intuition, a new c value can be computed using Equation 2.7.

$$c^* = (c + 1)\frac{N_c + 1}{N_c} \tag{2.7}$$

Using Equation 2.7, all the frequencies c are changed to a new count, c*, which is an adjusted count less than the original c. The probability of unseen features, count zero N_0, is computed

using the following equation:

$$P^*_{GT}(things\ with\ frequency\ zero\ in\ training) = \frac{N_1}{N} \tag{2.8}$$

Good-Turing causes a problem if the N_c+1 is zero. Thus, the values of N_c cannot used because they causes a problem if the N_{c+1} is zero. Therefore, the values of N_c need to be smoothed. One way to resolve this issue is to replace all N_c values with new values computed using linear regression, as seen in Equation 2.9, which maps N_c values. More details are discussed in reference [6].

$$log(N_c) = a + b\ log(c) \tag{2.9}$$

2.4.2 Maximum Entropy

The basic principle in maximum entropy is that when nothing is known, the probability distribution should be as uniform as possible, and the distribution is updated as evidence becomes known [7]. For example, considering an eight-way classification task, the probability of each class is 0.125 when nothing is known, as shown in Figure 2.3.

p(Bulgarian) + p(Chinese) + p(Chinese) + p(Czech) + p(French) + p(Japanese) + p(Native) + p(Russian) + p(Spanish) = 1

p(Bulgarian) = 1/8
p(Chinese) = 1/8
p(Czech) = 1/8
p(French) = 1/8
p(Japanese) = 1/8
p(Native) = 1/8
p(Russian) = 1/8
p(Spanish) = 1/8

Figure 2.3: Probability distribution without constraint

However, if there is evidence that would increase the likelihood of a particular class, the probability distribution would be updated accordingly. Suppose there is a feature that occurs in either *Bulgarian* or *Czech* 50% of time, we could apply this knowledge to update our model by requiring that *p* satisfy two constraints:

There are many probabilities that satisfy the two constraints, but the reasonable choice of *p* is

p(Bulgarian) + p(Czech) = 5/10

p(Bulgarian) + p(Chinese) + p(Chinese) + p(Czech) + p(French) + p(Japanese) + p(Native) + p(Russian) + p(Spanish) = 1

Figure 2.4: Probability distribution with a constraint

the most uniform, the distribution which allocates its probability as evenly as possible, subject to the constraints as shown in Figure 2.5 [8].

p(Bulgarian) = 1/4
p(Chinese) = 1/4
p(Czech) = 1/12
p(French) = 1/12
p(Japanese) = 1/12
p(Native) = 1/12
p(Russian) = 1/12
p(Spanish) = 1/12

Figure 2.5: Probability distribution without constraint

Let us discuss the concept of Maximum Entropy mathematically. Maximum Entropy uses the training data, *D*, which is a collection of contexts in documents *d* from all classes *c*, and uses *D* to construct a classifier via the conditional distribution *p* to classify "class" *c* given some "context" *d* based on the evidence from *D*. As shown above, the evidence allows us to set constraints that identify a set of feature functions that will be useful for classifying and measuring its expected value. These constraints can be written in the form of functions of contexts in documents and the class $f_i(c, d)$. Maximum Entropy combines constraints by assigning weights to the features using a exponential model:

$$p(a|b) = \frac{1}{Z(b)} \prod_{j=1}^{k} \alpha_j^{f_j(a,b)} \tag{2.10}$$

$$Z(b) = \sum_{a} \prod_{j=1}^{k} \alpha_j^{f_j(a,b)} \tag{2.11}$$

where Z(b) is a normalization factor to guarantee $\sum_a p(a|b) = 1$, and *k* is the number of features. Each parameter α_j corresponds to one feature f_j and also known as "weight" for that feature.

As discussed above, Maximum Entropy allocates probability distribution as evenly as possible, so it computes the entropy of all conditional probabilities and finds the most unconstrained distribution, *p**, using the following equations, which is the log of the Equations 2.10 and 2.11:

$$H(p) = -\sum_{a,b} \tilde{p}(b)p(a|b)\ log\ p(a|b) \tag{2.12}$$

$$p^* = argmax_{p \epsilon P}\ H(p) \tag{2.13}$$

$H(p)$ denotes the conditional entropy averaged over the training set, and $\tilde{p}(b)$ is the observed probability [9].

2.5 Evaluation

There are several different ways to measure results, and different measurements indicate success in different aspects of a given problem. This section presents the evaluation metrics that were used in this research.

2.5.1 Precision and Recall

Precision measures the correctness of the measurement by measuring the proportion of correctly classified items from the total number of items that were classified as the targeted class.

In other words, using the data in Table 2.2 as an example, precision for Native measures how many times a document was correctly classified as Native out of all cases where items were classified as Native (summation of the Native column); using Table 2.2, Equation 2.14 shows

	Classified As								
Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	12.7	0.7	0.7	0.9	2.2	0.5	0.9	1.4	20.0
Bulgarian	0.1	14.0	0.4	0.7	1.3	0.4	1.9	1.2	20.0
Chinese	0.8	0.3	16.2	0.5	0.5	1.1	0.6	0.0	20.0
Czech	0.7	1.4	0.5	11.4	1.4	0.6	3.6	0.4	20.0
French	1.6	1.3	0.1	1.1	11.2	0.1	2.0	2.6	20.0
Japanese	0.8	0.3	1.3	1.1	0.6	15.1	0.4	0.4	20.0
Russian	0.5	1.8	0.4	2.7	1.9	0.6	10.5	1.6	20.0
Spanish	0.8	1.3	0.2	1.1	2.2	0.1	1.4	12.9	20.0
Total	18.0	21.1	19.8	19.5	21.3	18.5	21.3	20.5	160.0

Table 2.2: Confusion Matrix

the computation.

$$Precision = \frac{12.7}{12.7 + 0.1 + 0.8 + 0.7 + 1.6 + 0.8 + 0.5 + 0.8} = 0.705 \tag{2.14}$$

Recall, on the other hand, measures the number of correctly classified items in relation to the total number of items that were categorized as a class, whether they are correctly classified or not. Again, using the data in Table 2.2 as an example, on average, Native classified 12.7 times as a true positive out of a total of 20 (summation of the Native row), which is the number of items that were classified as the Native class. Native recall is computed in Equation 2.15.

$$Recall = \frac{12.7}{20} = 0.635 \tag{2.15}$$

2.5.2 Accuracy

Another metric is accuracy. Accuracy measures the number of correctly classified items out of all cases. Another way to describe accuracy is measuring the degree of closeness from the true value. Using the data from Table 2.2, the accuracy for Native is computed by dividing the summation of data diagonally, from top left to bottom right, by the total number of cases, as shown in Equation 2.16.

$$Accuracy = \frac{12.7 + ... + 12.9}{160} = 0.65 \tag{2.16}$$

Precision cares only about how many of the items are correctly classified out of all items that were classified as the targeted class, so it does not say anything about true items that were not classified as true at all. In other words, precision just measures exactness. Recall, on the other hand, measures the number of correctly classified items in relation to the total number of items that were categorized as a class while disregarding items that were falsely classified as true. In other words, recall just measures completeness. For example, let's say out of 100 people, there are 20 terrorists. If the FBI captures 16 terrorist suspects and 13 of them are actual terrorists (true positive), then there are three innocent people who are captured (false positive), and there are seven terrorists who are not captured (true negative). Figure 2.3 shows the confusion matrix for this example. The precision of capturing terrorists is 0.8125. However, precision does not say anything about the other seven terrorists who were not captured. On the other hand, recall takes the terrorists who were not captured into account and computes as 0.65, but it does not take into account the innocent people who were captured. Accuracy measures take into account only people who are correctly classified out of an entire population while disregarding both false positives and true negatives, and it is measured as 0.9. The accuracy is misleading, since it seems to indicate that the FBI captured 90% of the terrorists, which was actually driven by the larger number who were not captured and are not terrorists (true negatives).

	Classified As	
Actual	Terrorist	Not Terrorist
Terrorist	13	7
Not Terrorist	3	77
Total	16	84

Table 2.3: Example

2.5.3 F-Score

The f-score is another way to measure accuracy by weighing both precision and recall. The f-score is the harmonic mean of recall and precision, which means it achieves a high value if both precision and recall values are high. If either recall or precision is low, the f-score will also be penalized. The f-score of the terrorist example is computed in 2.17.

$$F - score = \frac{2}{[\frac{1}{recall} + \frac{1}{precision}]} = \frac{2}{\frac{1}{0.8125} + \frac{1}{0.65}} = 0.72 \tag{2.17}$$

The f-score is a measurement that solves issues other measurements suffer by evenly weighing recall and precision and ignoring true negatives which influence the accuracy values.

2.6 Content Word Modeling

As part of an analysis, it is fair to investigate the contents in the corpus. A sub-corpus may include a particular word that is unique to that subcorpus. For example, if a significant portion of documents written by Chinese natives concern particular content words, those content words may significantly contribute to discriminating between the sub-corpora, which is not a desirable case in the nature of this research, since we want to exploit L1-L2 language transfer to build a model instead of a content-based model. Therefore, we employed two content-modeling tools to learn more about the documents in the corpus. The first tool we used is called Latent Dirichlet Allocation (LDA). LDA has the ability to generate a document using distribution over topics, in which each topic itself is a distribution over words. In other words, LDA can be used to test if each document in a subcorpus has similar topic proportions that are different from the topic proportions of other subcorpora. The second tool we used is called Term Frequency-Inverted Document Frequency (TF-IDF). TF-IDF has the ability to discriminate between content words that are unique to a document or subcorpus by assigning each word with a weight. If a particular word is unique to that document, TF-IDF gives a high score, but he words such as function words, which can be found in all documents, will receive very low scores. Therefore, TF-IDF can be used to remove content bias issues in this research.

2.6.1 Latent Dirichlet Allocation (LDA)

LDA is a generative system that builds a statistical model that can generate documents. LDA views each document as distribution over topics where a topic is modeled by a distribution over words. LDA uses these distributions to generate new documents using actual words. Let *n* be the number of words in a document. For each word slot, LDA selects a topic according to the topic distribution (θ_d) and the assigned topic is called *z* where *z* is equivalent to a β_k for some k. Since each topic is actually a distribution of words (β_k), each slot (*z*) will be filled by a word (w_i) in respect to β_k. Figure 2.6 graphically shows the relationship between these variables. To indicate that the system knows what w_i are, w_i are shaded, since words in documents are given. Therefore, distribution over words (β_k) and distribution over topics (θ_d) are not shaded, and they care called *hidden variables*. The variable α that points to θ_d in Figure 2.6 indicates that α actually generates θ. Although is it not shown in Figure 2.6, there is a variable called η which generates β.

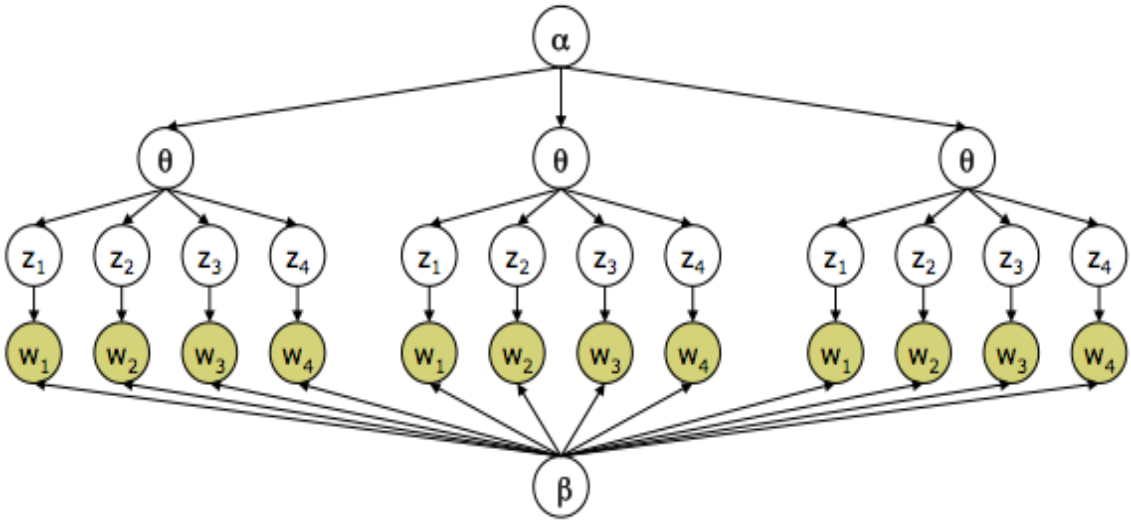


Figure 2.6: A graphical model representation of the latent Dirichlet allocation (LDA).

Figure 2.7 shows a simple example of how the LDA model generates a document when an input document consists of five words and assigned topics are two.

d_1 = 5 words in this document
k = 2 (there are 2 topics)
θ_{d_1} = { 0.8 (β_1) , 0.2 (β_2) }, distribution over topics in d_1
β_1 = $\{0.25_{w_1}, 0.25_{w_2}, 0.25_{2_3}, 0.25_{w_4}\}$ β_2 = $\{0.9_{w_1}, 0.1_{w_2}\}$

1) For d_1, there are 5 slots _ _ _ _ _, where each slot will be assigned with a topic z_i
2) d_1 becomes $\underline{z_1}$ $\underline{z_2}$ $\underline{z_3}$ $\underline{z_4}$ $\underline{z_5}$
3) Given the θ_{d_1}, d_1 likely to have $\underline{\beta_1}$ $\underline{\beta_1}$ $\underline{\beta_1}$ $\underline{\beta_1}$ $\underline{\beta_2}$
4) Since both β_1 and β_2 are given, word slots in d_1 will be replaced with the actual words with respect to β as follows: $\underline{w_1(\beta_1)}$ $\underline{w_2(\beta_1)}$ $\underline{w_3(\beta_1)}$ $\underline{w_4(\beta_1)}$ $\underline{w_1(\beta_2)}$
5) LDA will repeat the steps 1 through 4 until probabilities of selected words in each slot converge.

Figure 2.7: LDA example

2.6.2 Term Frequency-Inverted Document Frequency (TF-IDF)

As discussed above, the ideal corpus for this type of thesis is a content-free corpus; therefore, we used the TF-IDF technique to identify the content words. To state more accurately, TF-IDF does not know whether a word is a content word or not, but it detects all words that are unique to

a particular class. For example, if the word *Japan* shows up frequently in writings by Japanese, but rarely in other languages, TF-IDF weighs highly on the word *Japan*. The first term *TF* refers to the frequency of a term in a document, so if a term appears frequently in a document, the TF-IDF value will be high. On the other hand, the second term, *IDF*, refers to how frequently the same term appears in other documents. If the same term appears less frequently in other documents, than the value of *IDF* will be higher than when the term appears frequently in other documents as well. Equation 2.18 describes TF-IDF mathematically:

$$tfidf_t = f_{t,d} \times log\left(\frac{D}{f_{t,D}}\right) \tag{2.18}$$

where $f_{t,d}$ is the frequency of term t in document d, $f_{t,D}$ is the number of documents in which t appears, and D is the total number of documents in the collection [5] [10]. Therefore, the weight of the term $t \epsilon d$ will be maximal if the term *t* is common in d but not common in other documents. If the term t is common in d but also common in other documents, $log\left(\frac{D}{f_{t,D}}\right)$ will bring down the overall weights.

2.7 Tools

2.7.1 NPSML Tools

The Naval Postgraduate School (NPS) has built a set of in-house tools to facilitate running machine learning tools in its natural language processing lab. The tools are set up so that once the input is in the NPSML format, it can be converted to appropriate input format for all third-party machine learning tools. These tools are all available to the public via the Internet [11].

2.7.2 Maximum Entropy (GA) Optimization Package

The NPSML format can be easily converted to the Maximum Entropy (GA) Model (Megam) optimization package file format. Megam, the most used machine learning tool in this research, is publicly available via the Internet [12].

2.7.3 Stanford Parser

The Stanford Parser is used for parsing and POS tagging. The Stanford Parser takes a file input and produces parsed trees, POS tags, and dependency types from each sentence. The Stanford Parser package is also publicly available via the Internet [13].

2.7.4 LDA

We used the LDA open source tool written by David M. Blei in this research [14]. The NPSML format can be converted to the LDA input format via a tool that was built in the NPS NLP lab. The feature set for NPSML format must be the bag of words.

2.8 Prior Work

We have discussed how the L1-L2 language transfer can influence non-native speakers' writing style in English and have also discussed various types of stylometric feature sets that can discriminate the writing of one person from that of others. A lot of these stylometric features have been successfully used in authorship attribution problems, and Stamatatos describes why some of the widely used feature sets are performing well in reference [3]. The remainder of this section introduces the past research that is related to this research, reviews what feature sets are used, and observes how well some of the feature sets work.

2.8.1 Koppel

To the best of our knowledge, the first published work on automatically detecting an author's native language was done by Moshe Koppel in 2005. Koppel tried to identify an anonymous author's native language by exploring stylistic idiosyncrasies in the author's writing [15]. Koppel used the data from International Corpus of Learner English version 1, which is the previous version of the same data that were used in this research. Koppel considered sub-corpora contributed from Czech, French, Bulgarian, Russian, and Spanish. In each sub-corpus, 258 essays were used, and the length of each essay was between 579 to 846 words.

Koppel used a variety of stylistic feature sets such as function words, letter n-grams, and errors and idiosyncrasies [15].

1. **Function words**: 400 specific function words were chosen, but Koppel did not list which words were used.

2. **Letter n-grams**: 200 specific n-grams were chosen.

3. **Errors and Idiosyncrasies**: Koppel considered a range of spelling errors, neologisms, and Part-Of-Speech (POS) bigrams and narrowed it down to 185 error types and 250 rare POS bigrams as the feature sets.

Koppel used multi-class linear support vector machines (SVM) as the classification tool in his research, and with the chosen feature sets, he obtained 80.2% total accuracy classifying authors' native languages correctly, as shown in figure 2.8 [15] . The confusion matrix is shown in table 2.4. Koppel noticed that some features appeared more often in one class than in other classes [15]. Some of his observations are as follows:

- The POS pair most-ADVERB appeared more frequently in the Bulgarian corpus than in the other sub-corpora.
- A relatively large number of incorrect usages of double consonants was found in the Spanish corpus. Some specific errors were exclusively from the Spanish corpus, which could have been derived from the orthography of Spanish.
- Particular words, such as *indeed* and *Mr.* with a period, were frequently used in the French corpus.
- The Russian corpus was more prone to use the word *over* and the POS pair *NUMBER* more.
- The number of times the function word *the* was used per 1000 words: Czech 47, Russian 50.1, Bulgarian 52.3, French 63.9, and Spanish 61.4.

	Classified As				
Actual	Czech	French	Bulgarian	Russian	Spanish
Czech	209	1	18	20	10
French	9	219	13	12	5
Bulgarian	14	8	211	18	7
Russian	24	8	24	194	8
Spanish	16	10	10	7	215

Table 2.4: Confusion Matrix

Koppel achieved overall 80% accuracy determining the native language of the authors. He assumed the proficiency of the writers to be consistent throughout the data and normalized the features by the lengths of the essays; however, he discovered that the Spanish corpus was more prone to errors than the Bulgarian. In order to build a more accurate model, he recommended that the features be normalized by the error frequency from the entire corpus.

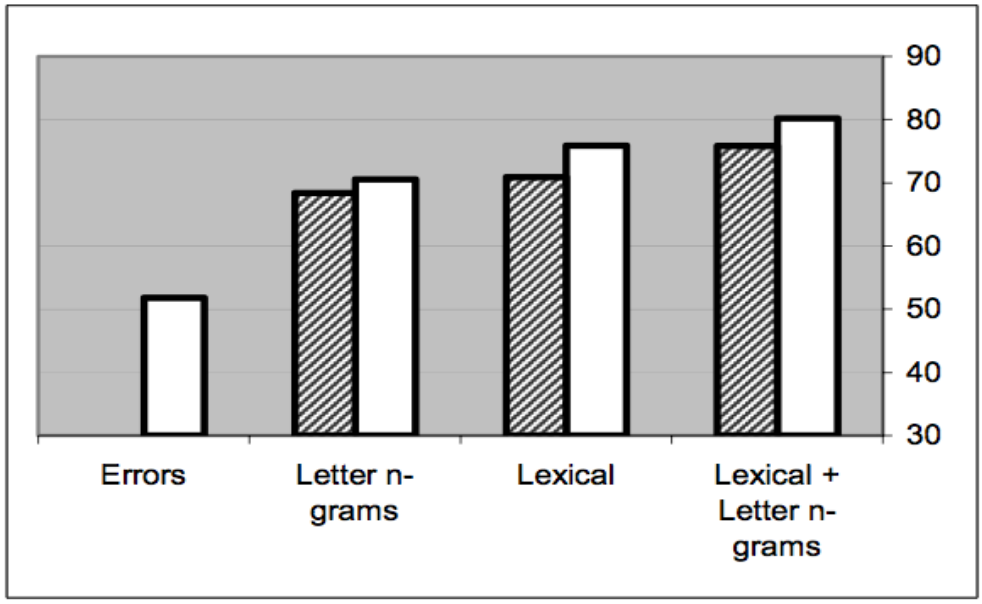


Figure 2.8: Accuracy (y-axis) on ten-fold cross-validation using various feature sets (x-axis) without (diagonal lines) and with (white) errors.

2.8.2 Rappoport

Ari Rappoport re-investigated Koppel's research on determining authors' native languages using character bi-grams as the only feature set. First, Rappoport chose the 200 most frequently used bi-grams in the whole corpus and used that as the feature set to achieve 65.6% accuracy with a standard deviation of 3.99 [12]. He then went further by choosing only the bi-grams that appeared at least 20 times in the whole corpus, or 84 bi-grams, and used only those bigrams to achieve a classification accuracy of 61.38%. Since bi-gram frequencies can be subject to content bias, Rappoport employed a statistical measure to evaluate and remove all dominant words in the sub-corpora and then repeated the classification experiments. The result was that the classification accuracy is essentially the same (it dropped only 2%). Rappoport also experimented with removing all the function words, to rule out the effect of the function words, and achieved 62.92% accuracy. Lastly, he replaced two of the sub-corpora, French and Spanish, with Dutch and Italian. With the new data set, Rappoport obtained 64.66% accuracy, essentially the same as in the original data set. Rappoport concluded that character bigrams may be capturing language transfer effects at the level of basic sounds and short sound sequences [10].

2.8.3 Wong

Sze-Meng Jojo Wong used Koppel's research on native language identification as a basis and investigated further by incorporating syntactic errors as an additional feature [16]. Wong used the latter version of the data set Koppel used with two additional sub-corpora: Japanese and Chinese. Wong selected three types of syntactic errors that non-native speakers were more prone to make as the features. The three selected syntactic error types are subject-verb agreement, noun-number disagreement, and misuse of determiners.

Wong conducted two different investigations using these syntactic features [16]. First, Wong observed the frequency of misuse of the three types of syntactic errors by all seven L1s and performed the classification tasks using only three syntactic errors as the feature sets. As shown in Table 2.5, the baseline is 14.29%, given that there are seven native languages with an equal number of data sets, and the classification accuracies are 5% higher with before-tuning and 10% higher with after-tuning. Wong used libSVM as the machine learning tool and a tool called Queequeg as the grammar checker.

Baseline	Presence absence	Relative frequency (before tuning)	Relative frequency (after tuning)
14.29% (25/175)	15.43% (27/175)	19.43% (34/175)	24.57% (43/175)

Table 2.5: Classification accuracy for error features

In the second part of Wong's investigation, she replicated Koppel's work and combined the replicated version of his work with the three syntactic features to investigate if integrating the three syntactic features improved the accuracy of 80%, which Koppel had achieved in his research. The classification results of these combined features are shown in Table 2.6. The best classification accuracy was achieved when function words and POS n-grams were used as the feature sets. Also, adding character n-grams as a feature set did not improve the accuracy.
Wong concluded that the three syntactic errors did not improve the overall accuracy because either not enough error types were used or the syntactic errors were not a good indicator for detecting an author's native language.

2.8.4 Summary of Prior Works

Koppel used four different types of feature sets to achieve total 80% accuracy when classifying the authors' native languages. Then Wong used Koppel's work as a baseline and investigated

Combinations of features	prior tuning (- errors)	prior tuning (+ errors)	after tuning (- errors)	after tuning (+ errors)
Function words + character n-grams	58.29% (102/175)	58.29% (102/175)	64.57% (113/175)	64.57% (113/175)
Function words + POS n-grams	73.71% (129/175)	73.71% (129/175)	73.71% (129/175)	73.71% (129/175)
Character n-grams POS n-grams	63.43% (111/175)	63.43% (111/175)	66.29% (116/175)	66.29% (116/175)
Function words + char n-grams + POS n-grams	72.57% (127/175)	72.57% (127/175)	73.71% (129/175)	73.71% (129/175)

Table 2.6: Classification accuracy for all combinations of lexical features

further whether using three different types of grammatical errors would improve the classification performance. Wong selected the following syntactic error types: subject-verb agreement, noun-number disagreement, and misuse of determiners. Wong's research showed that using these grammatical error types did not improve the performance either because grammatical error types are not a good indicator or because not enough error types were used. Lastly, Rappoport investigated why character bi-grams alone work well and concluded that they may be capturing language transfer effects at the level of basic sounds and short sound sequences.

2.9 Conclusion

In this chapter, we discussed concepts that are relevant to this research. We discussed some of the factors causing L1-L2 language transfer, and then discussed different types of features that serve as inputs to the two described machine learning algorithms for classification. We also discussed different metrics needed to evaluate the hypothesis. Lastly, we discussed prior research. We now have all the concepts and tools to design experiments to detect authors' L1 from their writing style.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Technical Approach

3.1 Introduction

In this chapter, the details of the experimentation setup are described. First, we describe the details of the corpus and how the data in the corpus are converted to a usable format for machine learning tools. Then, we continue with a discussion about the features selected for the experiments. Lastly, we present the details of the experimental setup for each machine learning technique.

3.2 Data Description

The data used in this research are from a corpus of academic (mainly argumentative) essays written in English by non-native speakers compiled by the project known as the *International Corpus of Learner English (ICLE)*. The length ranges mostly in from 500 to 1,000 words, and the authors are adults who are learning English as a foreign language but not necessarily as their second language [17]. The corpus has 16 different subcorpora categorized by the authors' native language. In this research, seven subcorpora are chosen: *Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish.* In addition to the data from the ICLE, a corpus called LOCNESS, which consists of native writings compiled by the *Centre for English Corpus Linguistics (CECL)* is also used as the eighth subcorpus. From each subcorpus, we selected 200 essays, each similar in size in terms of the number of words to maintain uniform size distribution of essays and eliminate the need for normalization. Tables 3.1 and 3.2 show the details of the size of each subcorpus in various ways. In general, essays in the Czech corpus tend to be longer than the essays in Chinese or Japanese corpora. Therefore, as Table 3.1 shows, the size of each subcorpus varies. For example, selecting the smallest size essays from the Czech corpus still makes the average Czech document size larger than the average document sizes of Chinese and Japanese. However, we believe that these size differences are not significant enough to affect the experiments, and therefore, we performed the whole experiment without normalization.

Class	Total number of words	Average number of words per essay	Total number of types
Bulgarian	145,412	727	9,581
Chinese	127,431	637	7,101
Czech	151,215	756	9,821
French	138,735	693	9,503
Japanese	120,152	600	8,047
Native	137,148	685	11,408
Russian	134,749	673	10,221
Spanish	129,951	649	10,291
Total	-	-	32,277

Table 3.1: Data size

Class	Total number of sentences	Average number of words per sentence
Bulgarian	7,206	20.2
Chinese	7,228	17.6
Czech	9,517	15.9
French	7,093	19.6
Japanese	8,590	14.0
Native	6,629	20.7
Russian	7,644	17.6
Spanish	5,858	22.2

Table 3.2: Number of sentences vs. average number of words per sentence

3.3 Raw Data to Usable Data

In the data from ICLE, each text file holds a written essay with a file name that reflects the author's native language and a unique number. Each essay originally had a unique identifier enclosed in brackets that held information on the author's native language, the institution the author belonged to when the essay was written, and a unique number. Since this information was inserted by a third person, it was removed. However, to preserve this information, the file name of each essay was named exactly as the unique identifier of that essay. Also, since all references were replaced by $<R>$, and all quotes were replaced by $<*>$, we removed all such indicators. Since this research was focused on the writings of non-native speakers, we decided to remove all special characters that were not written by the author.

Native writings from the LOCNESS corpus were compiled into a single file, but each essay was marked with the same type of identifier as the one we saw in the ICLE corpus. Therefore,

each native essay was saved as an individual file with a unique name. To maintain consistency, the file was named exactly like the unique identifier of that essay, as explained above. Also, as above, the native data had $<R>$ and $<*>$ as well, which were removed.

In addition to removing unnecessary content, all British spellings were replaced by U.S. spellings. We were able to identify a significant number of students who studied in Hong Kong and used British spelling on their essays. Therefore, to avoid British spelling being used as a discriminator, all British spelling was changed to U.S. spelling. We found a comprehensive list of words that were spelled differently in British in reference [18], and we used the list to replace all British spellings with U.S. spelling in the entire corpus.

3.4 Part-of-Speech (POS) Tagging

Once all the data were converted to usable data, we used the Stanford parser as a tool to generate part-of-speech (POS) tagging and phrase structure trees. The Stanford parser takes a text file as input as shown in Figure 3.1 and generates phrase structure trees with POS tags as an output.

```
./lexparser.csh inputFile.txt
```

Figure 3.1: Stanford parser command

For example, if the input file has a sentence *"Learning a new language is difficult!"*, the Stanford parser will generate the output as shown in Figure 3.2. If the input file has more than two sentences, the Stanford parser will automatically break them and produce an output per sentence. We parsed all 1,600 data files and piped the outputs into 1,600 new parsed files. Each parsed file was named by concatenating the actual data file name with the word *parsed* to easily keep track of all parsed files.

```
(ROOT
  (S
    (NP
      (NP (NNP Learning))
      (NP (DT a) (JJ new) (NN language)))
    (VP (VBZ is)
      (ADJP (JJ difficult)))
    (. !)))
```

Figure 3.2: Stanford parser output

3.5 Feature Extraction

As noted in the previous chapters, Koppel used character n-grams, part-of-speech n-grams, function words, and spelling errors as the feature sets to achieve 80% accuracy [15]. Therefore, we used Koppel's feature sets, with the exception of spelling errors, as a baseline to test our eight subcorpora. We also integrated the distribution of the transformation rule as a feature set to the baseline to test if the frequency of syntactic rules helps the overall performance. The list below shows the feature sets that were used at the beginning of the research.

- Character bigrams
- Character trigrams
- Character bigrams and character trigrams
- Function words (used a 456 function word list that was compiled independently)
- The top 200 POS bigrams (the top 200 most frequently-used bigram list was compiled from the NLTK Brown corpus)
- The top 200 POS trigrams (the top 200 most frequently-used trigram list was compiled from the NLTK Brown corpus)
- The top 200 POS bigrams & trigrams
- Function words, character bigrams and character trigrams
- Function words, character bigrams, character trigrams, the top 200 POS bigrams, and the top 200 POS trigrams
- Function words, character bigrams, character trigram, the top 200 POS bigram, the top 200 POS trigrams, and transformation rules

Rappoport states that a character bigram itself is an effective discriminator, so we tried with the higher character n-grams to test how output changed as the n-grams increased [10]. We also tried different variations of character trigrams as shown in the list below to test whether the performance of character trigrams' is affected by different features. For example, if running a character trigram on the corpus from which the punctuation has been extracted results in a

significantly worse performance than running a character trigram on the original corpus, we can learn that punctuation provides a significant indication to each calss and can be captured by a character trigram. The list below shows the list of the variations of character-trigrams we conducted.

- Character bigrams and up to character 7grams with case changed to upper case
- Character trigrams with no case change and extracted all trigrams that included spaces
- Character trigrams with no case change and no space (removed all spaces from the texts by combining each word with its adjacent words and then extracted character trigrams from those modified texts)
- Character trigrams with no case change, no space, and stemmed (used porter stemming to stem all words)
- Character trigrams with no case change and extracted all punctuation

We used the TF-IDF algorithm to identify content words as discussed in section 2.6.2 and re-built multiple versions of the corpora. The new corpora were differentiated by the number of content word types that were extracted, and Table 3.3 lists the number of types that were extracted in respect to the different thresholds. In other words, when the threshold for TF-IDF was set to 100, 245 word types were assigned with weights higher than 100, and these 245 word types were extracted to form a different corpus that has 245 fewer word types. Then the feature sets that are listed below were used on this new corpus. This approach allowed us to examine how performance changes as more content words, as identified by the TF-IDF, are extracted from the corpus.

- LDA coefficients on all different versions of corpora
- Character trigrams, upper case, no space, and stemmed on all different versions of corpora
- Character trigrams, upper case, no space, stemmed, and extracted all function words on all different versions of corpora
- Character 4grams, upper case, no space, and stemmed on all different versions of corpora
- Bag of words on all different versions of corpora

Threshold	Number of word types above the threshold
100	245
90	294
80	386
70	518
60	681
50	925
40	1292
30	1884
25	2358
20	3007
15	4052
10	6069

Table 3.3: Data size

3.6 NPSML Format

For each feature set, the feature was extracted from the eight subcorpora, and then the extracted data was formed into the NPSML format as shown in Figure 3.3. The first key field is the essay's class name and a unique number that distinguishes one essay from other essays within the class. For example, we named the Chinese essays from Chinese_1 to Chinese_200 as their unique ID. The second field, weight, is set at 1.0 for all cases. The class field is where we put the class name. The rest of the fields are feature labels and their counts.

key weight class feature_label_1 feature_value_1 feature_label_2 feature_value_2 ...

Figure 3.3: Feature extraction file format

Figure 3.4 shows a part of the NPSML format using character trigrams as the feature set. For every feature set, the respective features were extracted from the entire corpus and put into a form such as the NPSML format.

Bulgarian_1 1.0 Bulgarian all 3 rol 1 rom 3 ron 1 ali 2_us 3 osp 1 sca 1 lly 1 esc 1 _un 2 ...

Figure 3.4: Feature extraction file example for character trigrams

3.7 Initial Cross Validation

Once the features were extracted and converted to an NPSML format, the next step was to divide the data into test and training sets. The NPSML format was internally shuffled prior to creating test and training sets. We used ten-fold cross validation, which means ten different training and test sets were created by segmenting the data into ten different subsets. Then nine of these subsets were assigned as training data and the remaining subset was assigned as test data. Since there were ten subsets, ten different pairs of training and test data were formed. In creating models for each machine learning technique, training sets were used to train each model. Then, each model was tested with the test data.

3.8 Classification Tasks

Since our problem entails predicting an author's native language among the eight different classes, we performed multi-class classification tasks. After ten fold cross validation generated pairs of ten training and testing data, we used the training data to build a model using machine learning techniques, and when a model was built, we used the testing data, the pair of training data that were used to build the model, to perform the classification task. Then we used the result to build a confusion matrix. When all ten confusion matrices were generated, we averaged them out to create one final confusion matrix for each feature set as shown in Table 3.4.

	Classified As								
Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	12.7	0.7	0.7	0.9	2.2	0.5	0.9	1.4	20.0
Bulgarian	0.1	14.0	0.4	0.7	1.3	0.4	1.9	1.2	20.0
Chinese	0.8	0.3	16.2	0.5	0.5	1.1	0.6	0.0	20.0
Czech	0.7	1.4	0.5	11.4	1.4	0.6	3.6	0.4	20.0
French	1.6	1.3	0.1	1.1	11.2	0.1	2.0	2.6	20.0
Japanese	0.8	0.3	1.3	1.1	0.6	15.1	0.4	0.4	20.0
Russian	0.5	1.8	0.4	2.7	1.9	0.6	10.5	1.6	20.0
Spanish	0.8	1.3	0.2	1.1	2.2	0.1	1.4	12.9	20.0
Total	18.0	21.1	19.8	19.5	21.3	18.5	21.3	20.5	160.0

Table 3.4: Confusion Matrix

3.8.1 Naive Bayes

Naive Bayes experiments were conducted using a Naive Bayes package developed in the Naval Postgraduate School (NPS) natural language processing lab. The learning portion of this pack-

age uses an NPSML file as input and generates a model. The learning portion implemented the Good-Turing smoothing technique. The classification portion of this package used the model generated from the learning process to classify testing file, which is also NPSML file. The resulting output was a two column test file listing the key, the first column in NPSML format, and the predicted class.

3.8.2 Maximum Entropy

We used the Maximum Entropy GA Model (MegaM) Optimization package developed at the University of Utah to conduct experiments using maximum entropy [12]. NPSML files can be converted to MegaM format by removing the first two columns (key and weight). The learning portion of this package used a MegaM file as input and generated a model which by default was written to the standard output. The standard output was then piped to a file. The resulting model was a two column text file listing features and their weights. When running the learning portion of these experiments, the following command was used:

megam -quiet -nc -fvals -repeat 100 multiclass train.i > weights.i, where i is an index

Figure 3.5: MegaM command

The -quiet flag suppresses output to the screen. The -nc flag indicates that the names of classes are in text. The -fvals flag signifies the use of named features as opposed to an integer index to a feature list. The -repeat flag ensures that iterative improvement is attempted at least 100 times; this is needed to prevent the algorithm from stopping prior to convergence. The multiclass flag indicates what type of model to build.

3.9 LDA

We used the LDA open source code developed by David M. Blei to run LDA experiments. NPS developed an in-house converter that takes the NPSML format and converts it to the LDA input format. Once the LDA input format is prepared, the below command is used to generate topic models via LDA.

lda est 1.0 50 settings.txt mlformat.txt.lda random k_50

Figure 3.6: LDA command

The numeric value 1.0 indicates an initial alpha value and the value 50 indicates the number of topics, which is assigned by users. Settings.txt is the name of a file that holds the values of the parameters, and mlformat.txt.lda is the name of input file. The term random indicates that the topic will be initialized randomly. Lastly, k_50 is just a name of folder where all the models will be saved.

3.10 Conclusion

This chapter has presented a description of the data, the process associated with converting the data to usable data for machine learning, the features selected for the experiments, and the details of the experimental sets for each learning technique.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Results and Analysis

4.1 Introduction

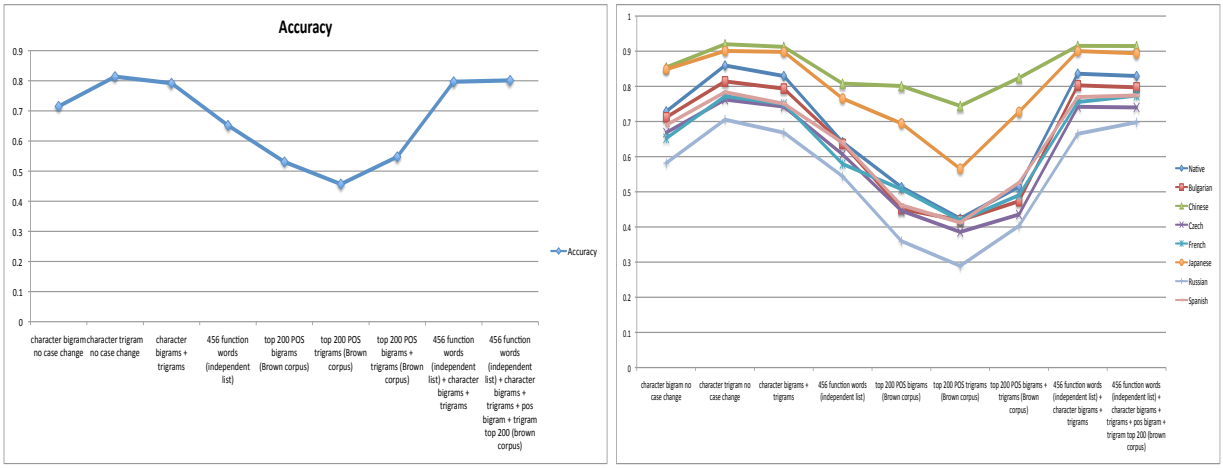
In this chapter, we present the results of our experiments as well as a discussion of their significance. We will first begin by discussing the results of the various feature sets and comparing the performances of maximum entropy and Naive Bayes classifications. We will then analyze the character trigrams and study what drives their success by empirical analysis, followed by a review of the performance of a particular lexical feature and character n-grams and their relationships. Lastly, we will discuss the role of topics in discriminating between authors and how the results change when topics are controlled.

4.2 Initial Classification Results

As discussed in Chapters 2 and 3, we used Koppel's feature sets, with the exception of spelling errors, as our baseline; Table 4.1 presents the results when maximum entropy was used as a classification method, and the results are plotted in graphs as shown in Figure 4.1.

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CB	0.715	0.728	0.711	*0.854*	0.668	0.652	0.848	0.581	0.689
CT	**0.813**	**0.859**	**0.814**	***0.919***	**0.762**	0.771	**0.900**	**0.705**	**0.783**
CBT	0.791	0.829	0.793	*0.912*	0.742	0.750	0.897	0.668	0.750
FW	0.651	0.641	0.636	*0.807*	0.606	0.579	0.765	0.544	0.639
POSB	0.530	0.513	0.451	*0.800*	0.446	0.507	0.694	0.359	0.461
POST	0.456	0.423	0.4189	*0.744*	0.385	0.4183	0.565	0.289	0.412
POSBT	0.547	0.514	0.473	*0.824*	0.435	0.490	0.727	0.402	0.526
FW CBT	0.796	0.835	0.802	*0.915*	0.741	0.755	**0.900**	0.665	0.769
FW A BT	0.801	0.829	0.797	*0.914*	0.740	**0.773**	0.894	0.697	0.774
CB	Character bigrams								
CT	Character trigrams								
CBT	Character bigrams & trigrams								
FW	Function words								
POSB	Top 200 POS bigrams								
POST	Top 200 POS trigrams								
POSBT	Top 200 POS bigrams & trigrams								
FW CBT	Function words and character bigrams & trigrams								
FW A BT	Function words and character bigrams & trigrams and top 200 POS bigrams & trigrams								

Table 4.1: Accuracy and f-scores across the feature sets



(a) Overall accuracies across the feature sets

(b) F-scores for individual language

Figure 4.1: Accuracies and f-scores across the feature sets

The highest value from each row is italicized, and the highest value from each column is in bold. As discussed in Chapter 3, the top 200 frequently-used POS bigrams and trigrams were compiled from the NLTK Brown corpus, and the list of 456 function words was compiled from independent sources. The following are some of our observations (see Appendix A for full confusion matrices):

- Koppel achieved 80.2% overall accuracy using the following feature sets: function words, character bigrams and trigrams, POS bigrams and trigrams, and spelling errors. We also achieved 80.1% overall accuracy using the same feature sets with the exception of spelling errors.

- Both character n-grams outperformed the other feature sets, and the character trigrams alone outperformed the results achieved from feature sets that combined character n-grams, function words, and POS n-grams. As discussed in Chapter 2, character n-grams capture nuances of style including lexical information, hints of contextual information, and use of punctuation and capitalization; we will show what is driving the character trigrams' high results in a later section.

- Function words are the only lexical level feature used in the initial experiments. Considering the dimension size which is much smaller than the character n-grams' dimension size, function words alone achieved 65% overall accuracy, which is higher than the per-

formance of POS n-grams. As we discussed in Chapter 2, some function words, such as prepositions, have specific syntactic functions governed by grammatical rules, while other function words, such as determiners, carry important semantic information. Therefore, it is reasonable to hypothesize that L1-L2 language transfer is a significant factor that drives the function words' performance. Table 4.2 shows the list of some of the most distinctive function words, which was determined by computing the entropy of each word and extracting the words with low entropies. Entropy measures randomness; in this case, entropy is high if the distribution of word usage is uniform and low otherwise. As expected, present-tense be verbs such as *am*, *is*, and *are* have high entropies and are not listed in Table 4.2, but we can also see that many pronouns and past-tense be verbs such as *I*, *you*, *she*, *were*, and *was* all made it to the low entropy list.

- Part of Speech (POS) bigrams and trigrams did not perform as well as the others. POS n-grams are syntactic features that capture authors' unique syntactic patterns. As discussed in Chapter 2, different languages have different word orders and use different branching directions. Although the overall performance of POS n-grams was not very good compared to the other results, the fact that Chinese and Japanese achieved high f-scores could indicate that their unique syntactic patterns are caused by their languages' grammatical distance being farther from the rest of the group. Additionally, we also used another syntactic feature called distribution of transformation rules, but it turned out that when this feature set was used alone, it performed poorly and when it was combined to the other feature sets, it dragged down the over-all performance. The result for the distribution of transformation rules is shown in Appendix C.

- The results of the initial experiments demonstrate that the Chinese f-scores outperform all other languages in all feature sets, and Chinese f-scores tend to fluctuate less across the different feature sets.

Function words with low entropies								
Function words	Bulgarian	Chinese	Czech	French	Japanese	Native	Russian	Spanish
ACCORDING	58	**248**	54	56	43	35	40	47
ALL	**709**	268	639	632	289	430	657	593
AMOUNT	26	**75**	32	20	10	57	27	20
ANYTHING	52	13	**83**	16	23	35	40	34
EVERYBODY	52	5	**95**	61	13	10	34	68
EVERYTHING	110	10	1**58**	73	23	40	121	68
FURTHER	37	22	9	**44**	10	38	22	9
FURTHERMORE	**26**	21	2	11	9	7	1	19
HE	238	97	**830**	630	384	632	488	310
HER	68	31	**281**	206	196	271	161	112
HIM	47	10	**145**	126	71	92	133	46
HIMSELF	19	7	37	**50**	7	35	39	15
HIS	228	55	**560**	413	186	479	403	209
HOWEVER	193	**295**	66	94	177	250	43	95
I	1000	470	1224	480	**2367**	542	942	547
INDEED	26	10	2	**116**	9	26	7	9
MAY	173	**562**	119	196	204	208	179	103
MOREOVER	54	64	10	**84**	32	4	18	39
MY	292	108	273	76	**538**	124	162	147
NOWADAYS	114	46	56	90	16	17	96	**144**
OUR	**905**	212	762	444	335	255	717	517
REAL	**210**	39	142	116	25	42	123	102
SHE	66	29	**308**	302	248	215	149	147
SOMETHING	**212**	33	190	71	77	73	144	154
UPON	32	3	12	15	10	**40**	39	12
VARIOUS	42	17	**67**	24	48	30	32	3
WAS	288	138	731	359	577	**680**	479	441
WE	54	10	42	154	4	8	37	98
WERE	180	69	339	221	196	**356**	262	278
WHAT	**594**	138	487	405	334	305	378	403
YOU	**715**	187	514	370	440	279	429	354
YOUR	**191**	26	181	66	74	72	118	78

Table 4.2: Most distinctive function words

4.3 Comparison Between Maximum Entropy and Naive Bayes

Using the exact same feature sets, we repeated the experiments with Naive Bayes (Good Turing smoothing technique) as the classification method. Table 4.3 shows the results, which are plotted into a graph in Figure 4.2. As the graph clearly shows, Naive Bayes performs poorly with POS n-grams, not to mention French f-score almost hits the bottom; however, Naive Bayes performs well for function words.

Machine Learning Tool: Naive Bayes									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CB	0.607	0.595	0.637	*0.830*	0.515	0.436	0.8144	0.522	0.525
CT	0.550	0.542	0.608	*0.819*	0.418	0.426	0.761	0.420	0.396
CBT	0.563	0.548	0.615	*0.824*	0.429	0.411	0.775	0.461	0.441
FW	**0.619**	**0.620**	***0.656***	0.826	0.506	0.477	*0.836*	0.4908	**0.562**
POSB	0.303	0.340	0.404	0.329	0.297	0.009	*0.483*	0.038	0.0853
POST	0.341	0.384	0.389	*0.589*	0.219	0.090	0.449	0.222	0.090
POSBT	0.360	0.369	0.444	*0.592*	0.303	0.047	0.535	0.181	0.142
FW CBT	0.579	0.593	0.623	*0.830*	0.470	0.454	0.789	0.438	0.443
FW A BT	0.608	0.598	0.486	**0.837**	**0.620**	**0.559**	***0.862***	**0.531**	0.345
CB	Character bigrams								
CT	Character trigrams								
CBT	Character bigrams & trigrams								
FW	Function words								
POSB	Top 200 POS bigrams								
POST	Top 200 POS trigrams								
POSBT	Top 200 POS bigrams & trigrams								
FW CBT	Function words and character bigrams & trigrams								
FW A BT	Function words and character bigrams & trigrams and top 200 POS bigrams & trigrams								

Table 4.3: Accuracy and F-scores for each feature set (Naive Bayes)

Figure 4.3 graphically shows the direct comparison of accuracies between MegaM (maximum entropy) and Naive Bayes. It clearly demonstrates that MegaM outperforms Naive Bayes on all feature sets. The f-scores comparisons for each individual languages are given in Appendix B, and they show roughly the same pattern. Japanese function words perform slightly better on Naive Bayes, but on the whole, MegaM does better. Therefore, the remainder of the research will be done only using MegaM. We also tried the intra-regional classification by collapsing the data and grouping them by region, and repeated the classification task using the same feature sets used in this section; however, the overall performance did not increase. The full results are shown in Appendix D.

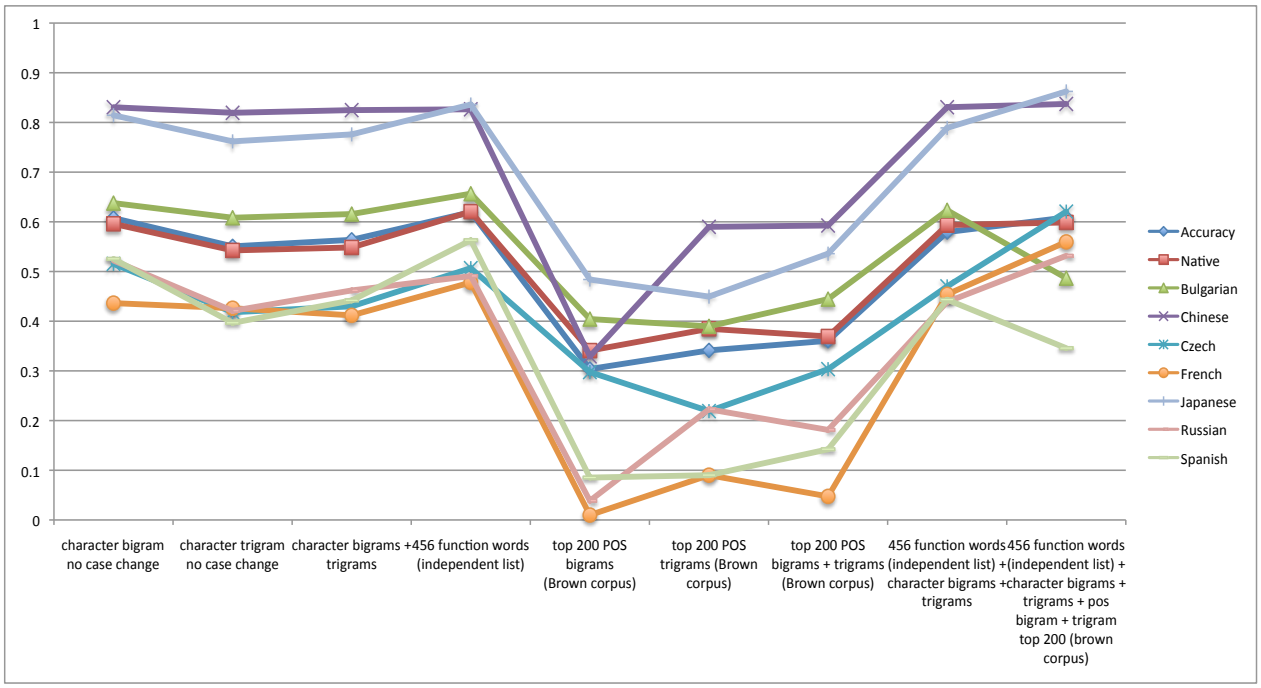


Figure 4.2: Accuracies and F-scores for individual languages (Naive Bayes)

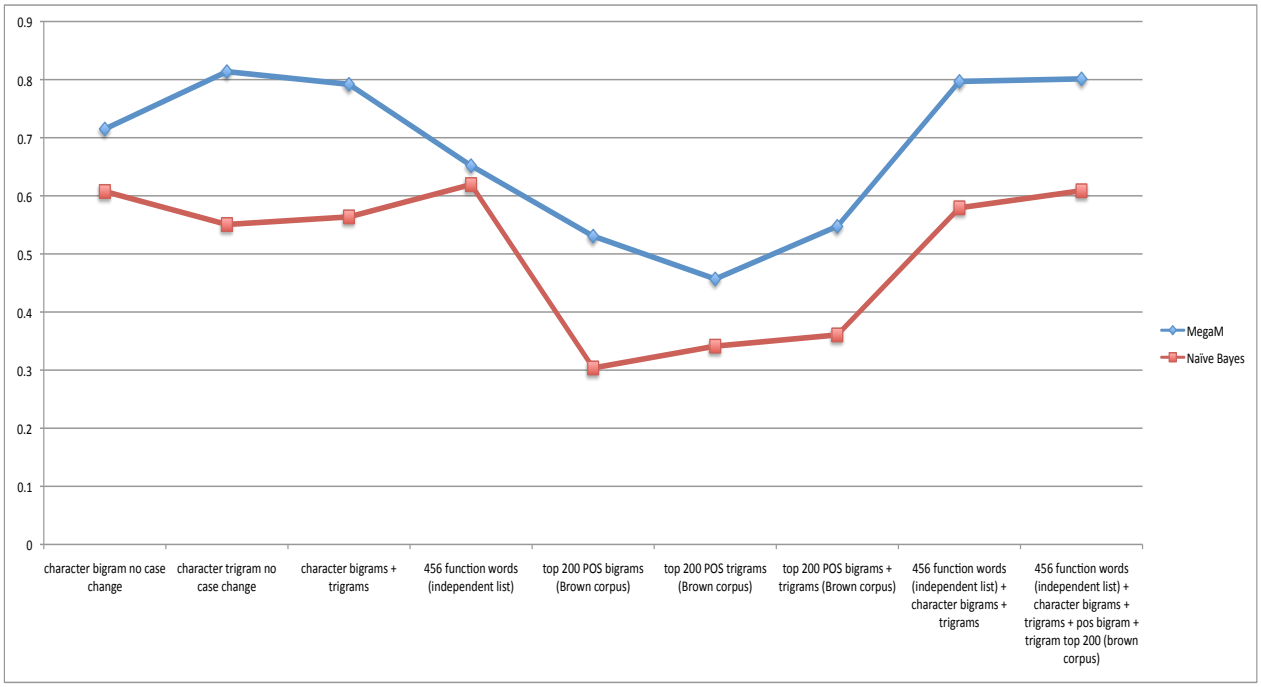


Figure 4.3: Comparing accuracies between MegaM and Naive Bayes

4.4 Character Trigrams Analysis

As shown above, character trigrams outperformed all other feature sets, and we discussed that character n-grams in general capture nuances of style, including lexical information, hints of contextual information, and use of punctuation and capitalization. To learn exactly what is driving the character trigrams' performance, we conducted several different types of character trigrams experiments as described in the following lists, and the results are shown in Table 4.4 and plotted in a graph in Figure 4.4.

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CT	**0.813**	**0.859**	0.814	***0.919***	0.762	0.771	**0.900**	0.705	**0.783**
CTU	0.807	0.851	**0.830**	*0.904*	0.761	0.764	0.889	0.686	0.779
CTGRW	0.798	0.817	0.804	*0.902*	0.755	0.763	0.879	0.704	0.770
CTNS	0.812	0.814	0.828	*0.896*	**0.773**	**0.789**	0.880	**0.750**	0.767
CTAS	0.800	0.820	0.800	*0.925*	0.761	0.744	0.882	0.713	0.763
CTNP	0.791	0.810	0.788	*0.902*	0.742	0.747	0.877	0.700	0.767

CT	Character trigrams
CTU	Character trigrams upper case
CTGRW	Character trigrams + no trigrams with white spaces
CTNS	Character trigrams + no white space
CTAS	Character trigrams + after stemming
CTNP	Character trigrams + no punctuation

Table 4.4: Accuracy vs F-scores

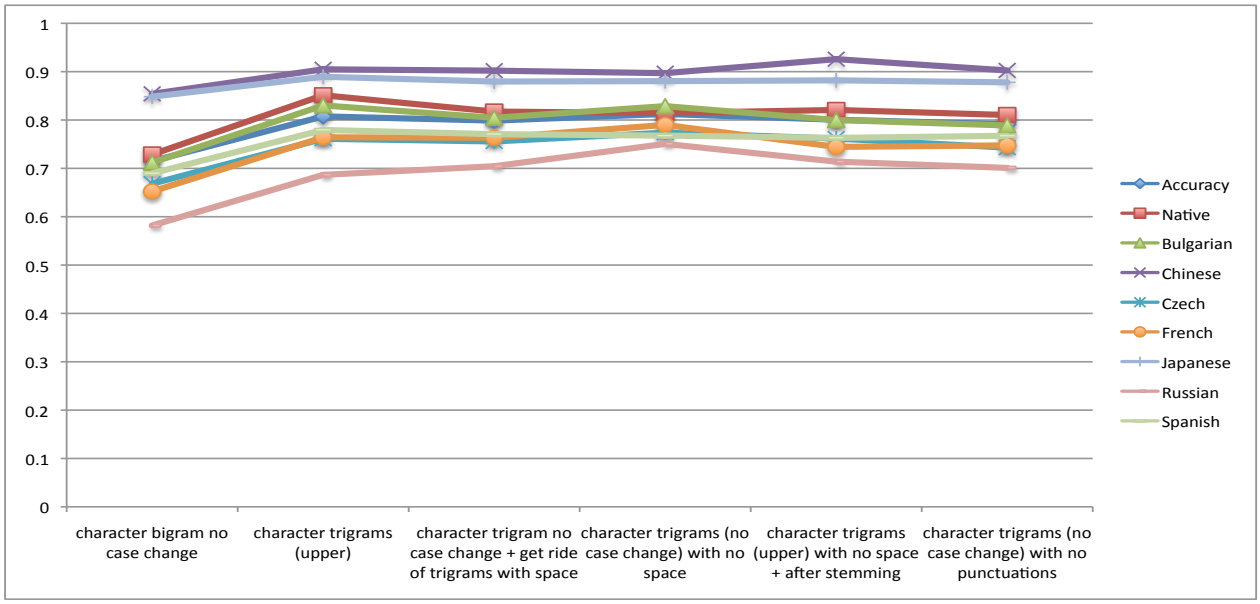


Figure 4.4: Character trigrams

- To eliminate the case bias, we changed the entire corpus' text case to upper case and repeated the character trigram experiments on the upper-cased corpus *(CTU)*. The result did not even drop 1%.

- To eliminate the space bias, we repeated the experiments without spaces using two different methods. In the first method, we removed all the character trigrams that had at least one space and performed the classification tasks with the rest of the character trigrams *(CTGRW)* . Tthe result shows that the overall accuracy dropped about 1.5%. In the second method, we removed the white spaces in the original corpus and performed the character trigrams classification task on the space-less corpus *(CTNS)*. The result also shows almost no changes.

- To eliminate the affix bias, we used the NLTK porter stemming tool, which converts the words to their root word, to stem all the words in the corpus, and we then repeated the experiments *(CTAS)*. The result shows a 1.3% drop.

- To eliminate the punctuation bias, we removed all the punctuation marks in the corpus and repeated the experiments *(CTNP)*. The result shows only a 2% drop.

We just learned from the experiments that case, spaces, affixes, and punctuation were not the major factors that affecting the character trigrams' performance. Therefore, in order to understand what drives the character trigrams' performance, we used the entropy techinque to identify the most distinctive character trigrams for further analysis; the most discriminating character trigrams are listed in Table 4.5.

In Table 4.5, underscores (_) indicate white spaces and the numbers to the right of the character trigrams' column represent the actual count of that particular character trigram seen in each language corpus. For example, the character trigram *hno* was seen 506 times in the Bulgarian corpus, but it was seen 104 times in the Chinese corpus.

Character trigrams with low entropies								
Character trigrams	Bulgarian	Chinese	Czech	French	Japanese	Native	Russian	Spanish
.__	2	70	17	32	3	1,835	5	3
_:_	4	3	28	422	0	12	49	16
_,_	46	70	30	55	14	36	413	109
_TV	38	182	362	65	73	3	248	56
...	97	7	77	243	8	15	68	173
:_t	26	3	15	190	4	18	59	87
_Br	27	70	19	59	22	344	26	32
hno	505	104	149	68	22	41	134	166
TV_	28	136	304	41	53	2	186	41
nol	506	105	150	70	26	46	136	168
Bul	102	0	0	0	3	9	2	0
uro	19	3	66	987	40	374	47	103
Bri	20	65	10	48	21	313	14	18
rop	118	98	146	1067	77	470	123	168
ngl	129	44	72	110	1062	267	82	59
_Mr	2	7	79	222	11	9	2	11
fes	123	571	94	62	32	77	176	187
rmo	53	33	19	226	16	25	26	33
van	108	509	72	87	67	68	51	155
apa	89	25	16	62	909	43	26	71
dva	68	496	49	59	61	56	40	115
_Ho	106	955	95	86	188	205	55	69
tag	63	543	81	87	61	90	55	93
ony	18	3	14	211	30	12	17	31
Eur	18	2	61	954	30	358	42	94
edi	105	960	139	106	122	181	150	157
.._	54	10	42	154	4	8	37	98
_Am	29	10	37	69	116	224	44	25
S_-	116	3	88	34	1	35	46	31
nty	18	10	38	36	16	163	37	40
_Ru	4	1	17	6	4	5	124	8
Ame	28	9	33	58	120	228	43	22
ewa	110	9	19	9	3	46	21	11
_Ra	9	9	65	197	8	38	6	10
e_-	115	7	100	32	3	14	41	17
rmy	1	32	107	42	5	7	192	131
_Sp	7	11	12	28	13	17	7	145
eam	992	16	558	195	62	95	415	262
ybe	57	413	53	27	39	28	18	49
deg	237	62	33	74	15	16	51	111
sov	1	3	7	16	5	136	14	16
_Ca	10	66	48	24	45	271	23	34
s_R	1	2	31	139	1	8	3	8
dit	86	970	106	94	147	129	124	103
ilm	34	5	120	40	5	5	117	49
ane	67	15	50	67	578	39	62	39
Int	37	43	3	7	132	10	14	20
_sm	65	714	99	53	115	87	63	50

Table 4.5: Character trigrams with low entropies

We chose six character trigrams from Table 4.5 and pulled out all the words that these six character trigrams are capture. In other word, we identified the original words that are originating these six character trigrams, and listed them in Tables 4.6 to 4.11. Table 4.6 shows that *hno* was mostly driven from the word *technology* and it was a good marker for Bulgarian; the Native corpus frequently used the word *British*, which was captured by *Bri* as shown in Table 4.7; the Japanese corpus frequently used the word *Japanese* and *Japan*, captured by *apa*; the Chinese corpus frequently used the word *professionals*, captured by *fes* and *Hong*, captured by *_Ho*; and the French corpus frequently used the word *harmony*, captured by *rmo*. Beside the six character trigrams we just presented, there are other distinctive character trigrams originated by content words: the character trigram *Bul* originated from the word *Bulgarian*, which was a good marker for Bulgarian; *rop* and *Eur* originated from the word *Europe*, which was a good marker for French; *_Sp* originated from the word *Spanish*, which was a good marker for Spanish. The most important finding from this empirical analysis is that most of the distinctive character trigrams are capturing the content words in the corpus.

If it is the case that character trigrams' performance is driven by the content words, it also explains why eliminating the case-sensitive, white space, punctuation, and affixes biases did not affect the overall accuracy significantly, as we saw from the previous sections. For example, when we moved spaces, the reason why the performance did not drop much was because the relevance trigrams in word boundaries are not themselves causing the discrimination. Let us say we have *"in the "* and *"in every"* in a sentence, and when we get rid of spaces, then we are left with *nth* and *nev*; it might be that those trigrams happen less frequently across the character trigrams then internal trigrams in a discriminative word such as *"technological"*. Therefore, the information that space provides in discriminability is much less than the character trigrams inside of the content words.

Trigram: hno	
Class	Words from the corpus
Bulgarian	technology (325), technological (99), technologies (51), Technologically (2)
Chinese	technology (88), technologies (4), high-technology(4)
Czech	technology (131), technological (8)
French	technology (51), technologies (7), technological (7)
Japanese	technology (15), technologies (4), thechnology (1), technorogy (1)
Native	technology (26), technological (5), ethno-centric (1), ethnocentric(1), biotechnology (1)
Russian	technology (95), technological (21), technologies (11), technocratic (1), technologisation (1)
Spanish	technology (134), technological (22), technologycal (2), thechnology (1)

Table 4.6: Actual words that includes the trigram "hno"

Trigram: Bri	
Class	Words from the corpus
Bulgarian	British (9), Britain (8), Brilliant (1)
Chinese	British (33), Britain (31), Bridge (1)
Czech	British (3), Britain (3), Brisc (2)
French	Briscoe (21), Britain (12), British (10), Great-Britain (4)
Japanese	British (12), Britain (7), Bright (1)
Native	Britain (187), British (100), Britons (11)
Russian	British (7), Britain (7)
Spanish	Britain (13), British (4)

Table 4.7: Actual words that includes the trigram “Bri”

Trigram: apa	
Class	Words from the corpus
Bulgarian	capable (28), apart (16), capacity (16), incapable (9)
Chinese	capacity (17), capable (5), Japan (1)
Czech	capable (7), apart (6)
French	capacity (13), Japan (13), capable (11), apart (11)
Japanese	Japanese (520), Japan (344), Japanes (8), capability (2)
Native	apart (12), capable (6), capacity (6), capabilities (5)
Russian	capable (8), apartment (4), capacity (3), apart (3)
Spanish	capacity (31), apart (13), capable (13), capacities (4), incapable (3)

Table 4.8: Actual words that includes the trigram “apa”

Trigram: fes	
Class	Words from the corpus
Bulgarian	professions (43), professional (40), , professors (10)
Chinese	professionals (235), cafes (212), professional (80)
Czech	professional (38), profession (10), confess (7), lifestyle (5), lifes (4)
French	professional (39), professors (7), lifestyle (1)
Japanese	professional (13), professor (5), festival (3)
Native	professors (21), professional (12), professions (9), lifestyles (6)
Russian	Professional (113), profession (33), lifes (6), professor (5)
Spanish	professional (95), lifes (55), professions (7)

Table 4.9: Actual words that includes the trigram “fes”

Trigram: rmo	
Class	Words from the corpus
Bulgarian	Furthermore (26), harmony (11), enormous (9)
Chinese	Furthermore (24), harmony (2), hormones (2), enormous (2)
Czech	enormous (8), harmony (5), Furthermore (2)
French	harmony (184), Furthermore (11), enormous (7), harmonizsation (6)
Japanese	Furthermore (9), harmony (4), enormous (3)
Native	Furthermore (6), enormous (3), harmony (2)
Russian	enormous (12), harmony (8)
Spanish	Furthermore (19), enormous (7), armory (2)

Table 4.10: Actual words that includes the trigram “rmo”

Trigram: (space) + Ho	
Class	Words from the corpus
Bulgarian	However (73), How (17)
Chinese	Hong (776), However (159), How (13), Hongkong (2)
Czech	How (42), However (39)
French	However (51), How (27)
Japanese	However (135), How (22), Hokkaido (13)
Native	However (131), House (11), Hoederer (9)
Russian	How (25), However (15), Hollywood (3)
Spanish	However (50), How (9)

Table 4.11: Actual words that includes the trigram “space + Ho”

4.5 Lexical Model

In this section, we will observe a word-based model and compare the model to the character trigrams' model to see how performance changes as the top content words are extracted.

4.5.1 Lexical Model vs Character Trigrams Model

In the previous section, we discussed that the strong signals in character trigrams are driven from the content words; therefore, we will compare the character-trigrams model to the lexical model. We used bag of words, also known as word unigrams, as our lexical feature set to build a lexical model. Table 4.12 shows the performances of both lexical and character trigrams models, plotted in a graph in Figure 4.5. Based on these performances, it appears that there is a strong correlation between these two models, which is consistent with the observations we made from the character trigrams' analysis section.

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
BOW	0.8143	0.838	0.805	0.934	0.741	0.775	0.911	0.706	0.811
CT	0.813 7	0.859	0.814	0.92	0.762	0.772	0.901	0.705	0.783

BOW	Bag of words
CT	Character trigrams

Table 4.12: Character trigrams vs Bag of words

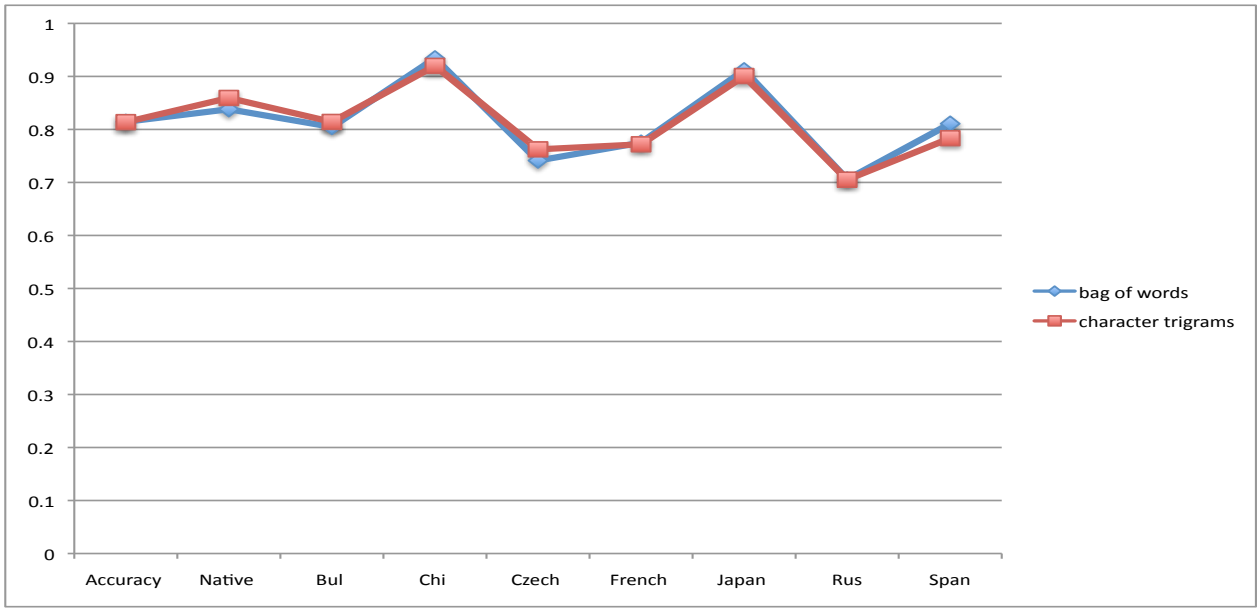


Figure 4.5: Accuracies: Bag of words vs Character trigrams

4.5.2 Latent Dirichlet Allocation (LDA)

We have seen that there is a strong correlation between the character trigrams and the lexical models; therefore, we hypothesize that the character trigrams simply simulate the lexical model. If the content words are doing all the work, it could be also seen as the distinctions between the documents written by different native speakers are actually driven by the topics. Therefore, we used the LDA model to verify this notion by using the distribution of topics as the feature sets, and results are shown in Table 4.13.

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
LDA coefficients	0.561	0.6103	0.675	0.823	0.371	0.567	0.762	0.308	0.346

Table 4.13: LDA coefficients (θ) (k-50)

Using LDA topic models across 50 topics alone as the feature set, we find that an indication that each language corpus contains a significant number of topic words that are unique to their corpus. The Chinese f-score is 0.823, which is significant considering that the vector space is only 50. The overall performance is not as high as some of the performances we saw in the previous sections, but this result is based on only 50 topics. We expect that as we increase the topic space to higher dimensions, the overall performance will increase respectively. For example, if Russians frequently used the word *computer* and Chinese frequently used the word *technology*, then in a small dimension topic space, the LDA model may cluster the two words into the same topic, but in a higher dimension space, there is a higher chance that these two words will be assigned to separate topics, which will help us to discriminate between Chinese and Russians. However, we feel that those experiments are unnecessary since the topic model with a dimension size of 50 has already shown that there are topics that need to be controlled for more precise experiments.

4.5.3 Results from a Topic-Controlled Environment

Above experiments with the LDA model demonstrate that topics are strong discriminative factors, and if topics are doing all the work, it is difficult to measure if there are other signals that may have been contributing to discriminating different languages. Therefore, we used Term Frequency-Inverted Document Frequency (TF-IDF) as the method to identify and extract the content words to control these topics. For simplicity, we will refer to topic related words as content words from now on.

$$tfidf_t = f_{t,d} \times log\left(\frac{D}{f_{t,D}}\right) \tag{4.1}$$

The first term, *Term frequency*, $f_{t,d}$, simply refers to the frequency of term *t* in document *d*, so if we are computing the word *technology*, $f_{t,d}$ would be the number of times the word *technology* appeared in a document. The second term, *Inverted Document Frequency*, $log(\frac{D}{f_{t,D}})$, refers to assigning higher weights to a word that occurs frequently in one document but less frequently in other documents. Therefore, if the word *technology* appeared in 35 different documents out of the entire corpus, IDF can be computed as $log(\frac{1600}{35})$ since there are 1,600 essays in our entire corpus. However, if the word *technology* appeared in 100 different documents, the IDF part will dampen the TF-IDF values. As discussed in the beginning of Chapter 3, we made each subcorpus size similar to maintain uniform size distribution of essays across the entire corpus and eliminate the need for normalization. Therefore, we did not normalize the values during the TF-IDF computation.

We extracted the top 114 words, top words in terms of the most weighted word by TF-IDF, and presented them in Table 4.14. In other words, the words in Table 4.14 are the words that were used frequently by writers of a particular language while infrequently used by those of other languages. The top weighted words include not only content words but also function words if the word counts meet the threshold. The words in bold are the ones we have seen from the previous sections when we analyzed the most discriminative character trigrams and what words those character trigrams were capturing. Table 4.14 also includes function words such as *I* and *You*, which is not very shocking since we have already seen these function words in Table 4.2, which listed the most discriminative function words.

THEORETICAL KNOWLEDGE STUDENTS DREAM **TECHNOLOGY** UNIVERSITY DREAMING EQUAL CONTRIBUTION WE DREAMS OUR EDUCATION IMAGINATION EQUALITY YOU SCIENCE INFORMATION SOCCER CARD WASTE TELEVISION **HONG** MAY ACCORDING SMOKE CARDS CAFES ADVANTAGES CYBER GOVERNMENT RESTAURANT BETTING CAFE MAINLAND SMOKERS MATERIALS BANNING SMOKING PLASTIC RECYCLING CHINA **PROFESSIONALS** RAILWAY ABORTION MANAGEMENT IMPORTING DEBT USE STUDENT DISADVANTAGES RESTAURANTS FINANCIAL KONG HEALTH INTERNET CREDIT WOMEN SCHEME USING LOCAL I CHILDREN SHE RELIGION TV HIS HER MONEY HE WAS **JAPANESE** MY DOG AINU PHONES E-MAIL DON'T **JAPAN** PENALTY CELL SCHOOL LANGUAGE CELLULAR PHONE SPEAK ENGLISH MRS NATION COUNTRIES DENIS **HARMONY** 1992 COMMUNITY EUROPE IDENTITY RAMSAY EUROPEAN MEN MILITARY SERVICE ARMY PRISON ETHNIC **BRITAIN** VOLTAIRE CANDIDE MARIJUANA BEEF LOTTERY SOVEREIGNTY BOXING SEX

Table 4.14: The top 114 most weighted TF-IDF words

To build a topic-free corpus, we have to extract all topic words; however, there is no definite number for how many words should be extracted. Thus, we used the 12 different TF-IDF thresholds as shown in Table 4.15, and conducted the experiments on all 12 cases. In other words, when the threshold was set to 100, TF-IDF removed the top 245 content words and then conducted classification tasking using various feature sets on the corpus that had 245 fewer words.

Threshold	Number of word types above the threshold	% of corpus
100	245	0.758 %
90	294	0.911 %
80	386	1.195 %
70	518	1.604 %
60	681	2.109 %
50	925	2.865 %
40	1,292	4.002 %
30	1,884	5.837 %
25	2,358	7.305 %
20	3,007	9.316 %
15	4,052	12.554 %
10	6,069	18.802 %

Table 4.15: Data size

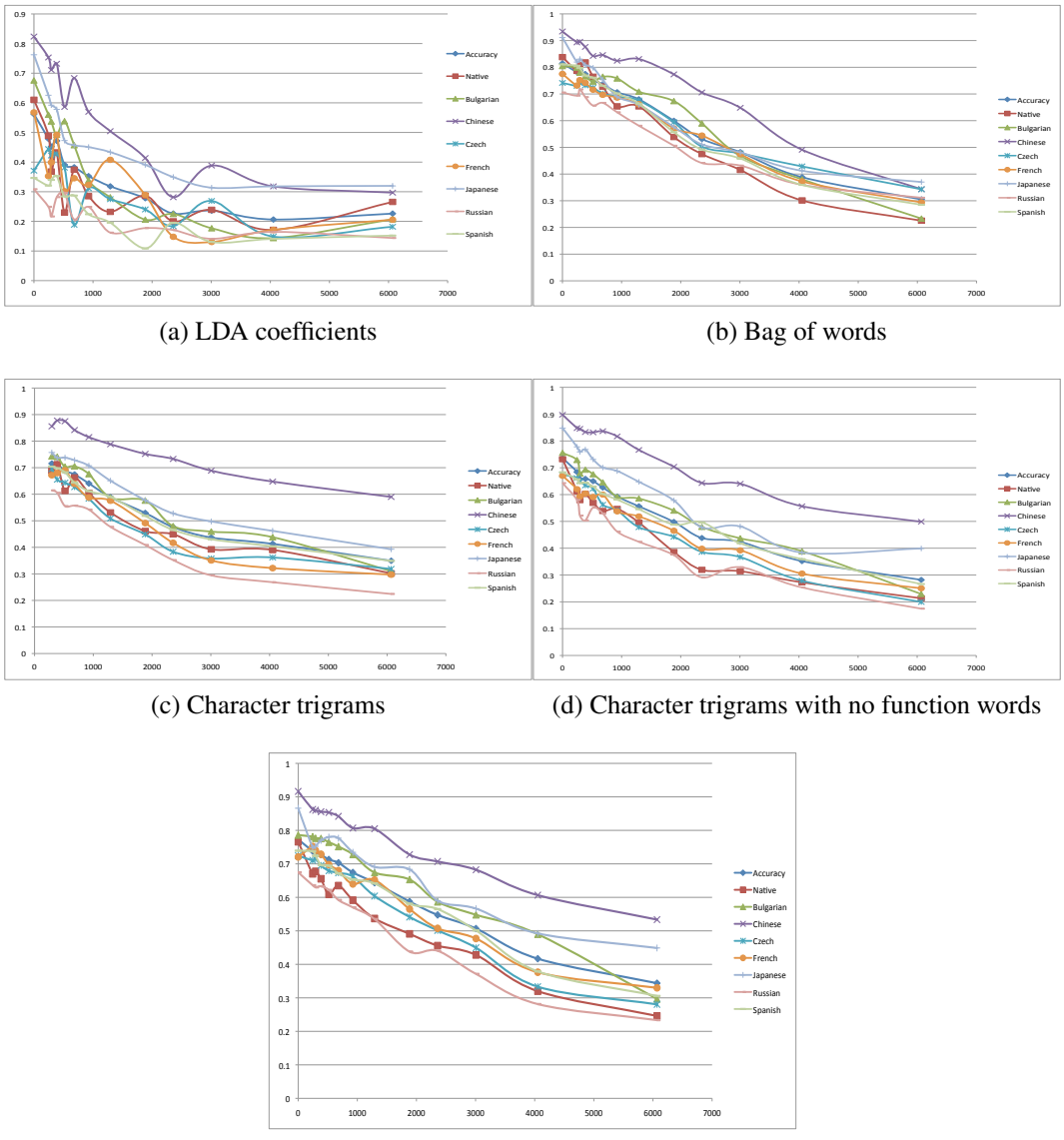
As discussed in Chapter 3, we used the following feature sets to experiment on the 12 different content-free corpora:

- LDA coefficients (k = 50)
- Character trigrams, uppercase, no space, and stemmed
- Character trigrams, upper case, no space, stemmed, and no function words
- Character quadgrams, upper case, no space, and stemmed
- Bag of words (word unigram)

For simplicity, in this section, when we say character trigrams, we are referring to character trigrams, upper case, no space, stemmed, and when when we say character quadgrams, we

are referring to character quadgrams, upper case, no space, and stemmed, and lastly, when we say character ngrams, we are just referring to the above-mentioned character trigrams and quadgrams.

Overall, when the most distinctive words are extracted, the performances dropped as expected. Figure 4.6 shows five graphs where each graph shows the performance of each feature set across the 12 TF-IDF threshold.



(a) LDA coefficients

(b) Bag of words

(c) Character trigrams

(d) Character trigrams with no function words

(e) Character quadgrams

Figure 4.6: Classification results after word extractions

The x-axis indicates the actual number of extracted word types, and the y-axis indicates measurement which is either accuracy or f-scores. In general, with the exception of the LDA model, the rest of the feature sets, subfigures 4.6b - 4.6e, show similar patterns while they are extracted. In the LDA model, the actual feature set is the distribution of topics, where each topic is a distribution of words. To recall, the dimension size of the LDA model is only 50, so performance is significantly worse than with other feature sets. Also, when we removed the top-content words, the distribution of words for each topic became more similar to other topic distributions, and when distinctions disappeared, it appears that random noises causing the oscillation that can be seen in the graph.

Figure 4.7 shows the direct comparison of accuracies between the character n-grams and the lexical model. The accuracy for the LDA model is not included since we are only interested in comparing the performances of bag of words and character n-grams.

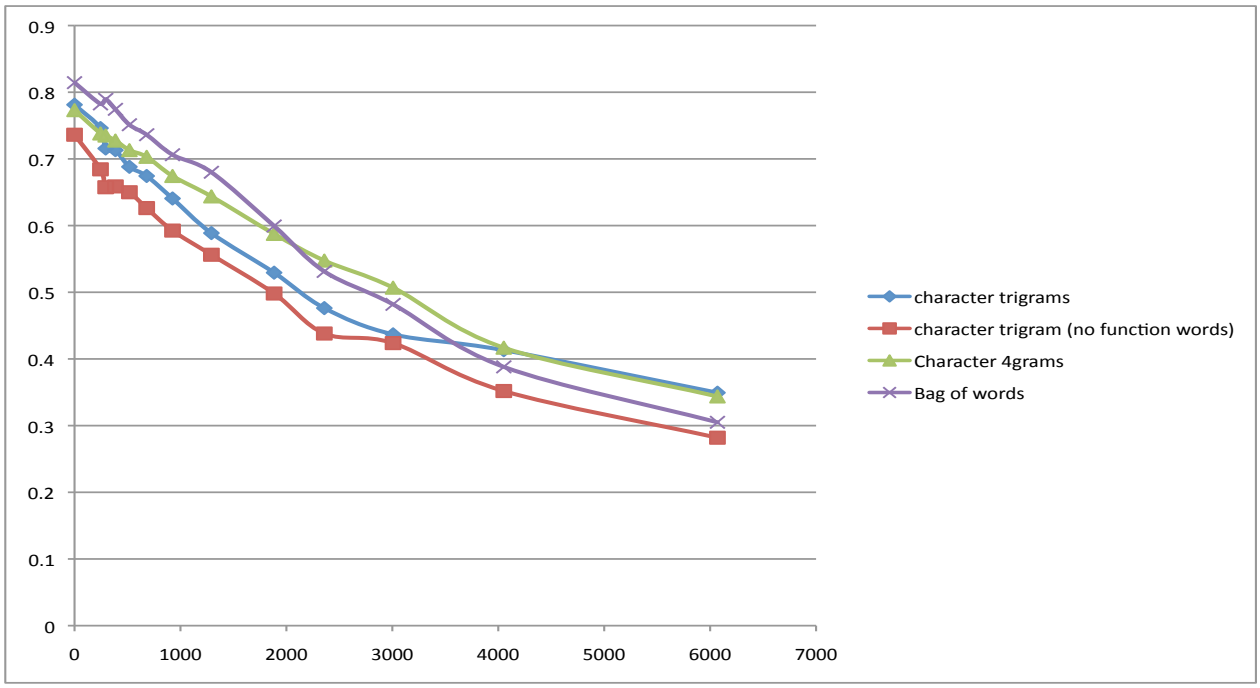
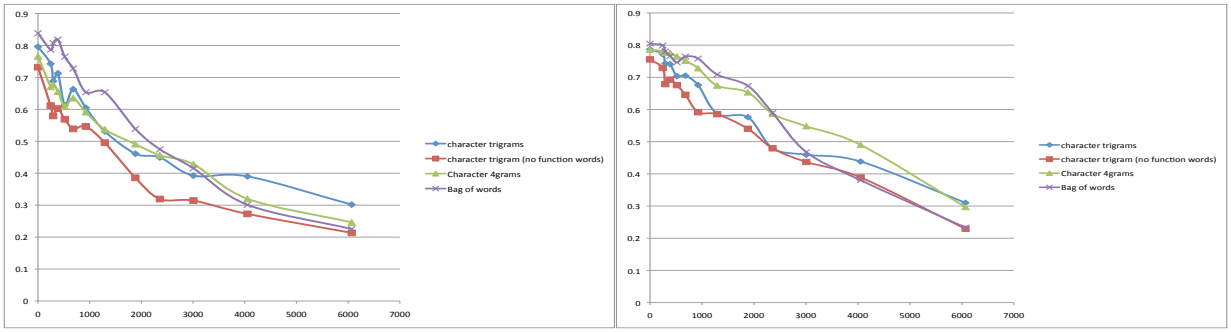


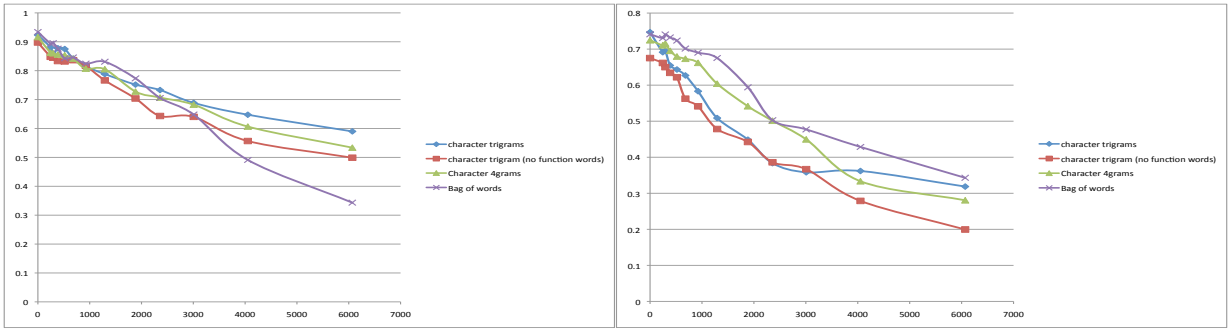
Figure 4.7: Accuracies

Figure 4.8 shows eight different graphs where each graph displays each language's f-scores for different feature sets.



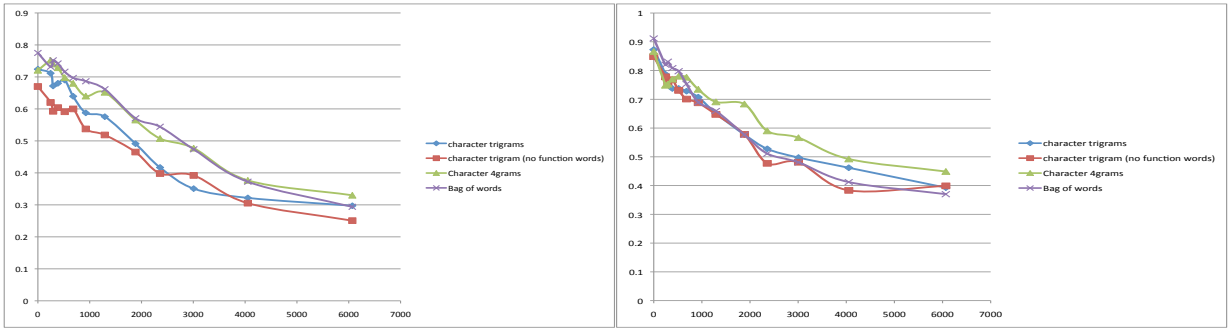
(a) Native f-scores

(b) Bulgarian f-scores



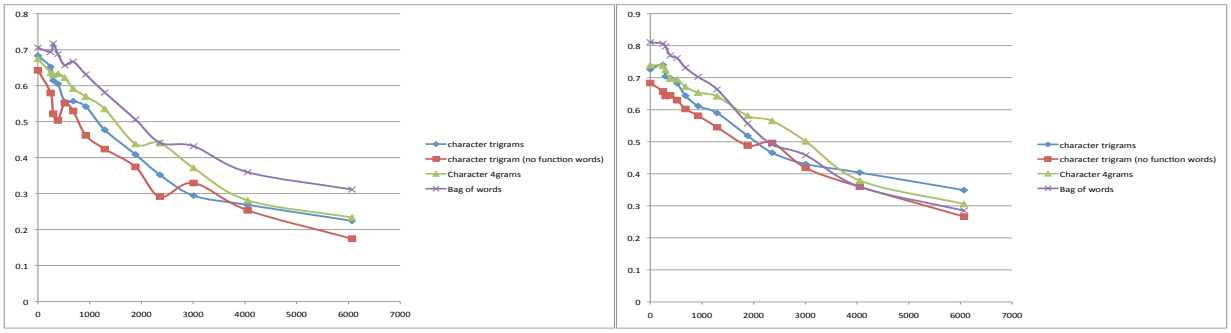
(c) Chinese f-scores

(d) Czech f-scores



(e) French f-scores

(f) Japanese f-scores



(g) Russian f-scores

(h) Spanish f-scores

Figure 4.8: Performances for individual languages

Based on these graphs, we made the following observations (tables of the complete results are shown in Appendix E):

- At first, the lexical model performed slightly better than the character n-grams, which suggests that although character n-grams are simulating the lexical model, as we discussed in the previous sections, they are not finely-tuned model since they also capture noises such as subsets of words that are adjacent to each other.
- When about 7% of the top words were extracted, the accuracy of the lexical model dropped below that of the character quadgrams, and when about 10% of the top words were extracted, it dropped below that of the character trigrams. When so many distinctive words are extracted, the word distributions become flat and the lexical model suffers more severely since there are fewer distinctions between the word distributions the lexical model has to rely on; however, character n-grams still capture other signals which are not relatively significant until the most discriminative character n-grams disappear due to content-word removal.
- The patterns of an individual language's f-scores are similar to the pattern of accuracy the most part. Each language has its own unique pattern to a degree, but the overall patterns are consistent.

4.6 Conclusion

In this section, we have initially explored the various types of feature sets that were explored in Koppel's research and learned that character trigrams are the best performing feature sets. After an empirical analysis of character trigrams, we also learned that they simulate the word model, which means that character trigrams just model lexical use. Then we used the LDA model to verify that the corpus contains unique topic distributions that may be influencing the performances of lexical and character models to a certain degree. To further investigate the relationship between character trigrams and the lexical model on a topic-free corpus, we used the TF-IDF techniques to extract the most distinctive words and repeated the experiments. As discussed in the latter section of the chapter, both character n-grams and bag of words performances dropped as the top words were extracted in similar patterns at first. However, when about 10% of word types were extracted, the lexical model's performance dropped more drastically than that of the character n-grams. It is difficult to measure how much the L1-L2 language transfer is influences lexical usage, but based on our experiments and results,

we conclude that the lexical model is the strongest feature set, and character n-grams simply simulate the lexical model until a significant amount of content words are extracted.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusions and Future Work

5.1 Summary

This thesis addressed three questions. The first question was, "How well can we detect an author's native language using various natural language processing tools?" As shown in Chapter 4, the answer is that we can detect authors' native language with a higher than 80% accuracy using either character trigrams or bag of words alone as a feature set. Syntactical feature sets such as POS n-grams and distribution of transformation rules worked fairly well for detecting Chinese and Japanese, but it performed less well with Slavics and Romance languages. We also compared the overall performance between Maximum Entropy and Naive Bayes, and the results showed that Maximum Entropy performed significantly better than Naive Bayes.

The second question was, "What is the strongest feature set and why does that particular feature set work better than the other feature sets?" The bag of words showed the best results, followed by character trigrams. Empirical analysis of character trigrams revealed that the most discriminative character trigrams originate in content words, which showed evidence that character trigrams are just modeling lexical usages. Based on these results, we concluded that the best indication for detecting authors' native languages is their lexical usage. There may be some signals caused by L1-L2 language transfer at the syntactic or character level, but if such signals exist, our hypothesis is that the frequency of the occurrences of these signals is significantly lower than that generated by the lexical feature sets, and they become insignificant.

The third question was, "To what extend is the second question dependent on the topics discussed in the corpus?" To answer this question, we used the LDA model to show that the distribution of topics of each language corpus is distinct from other distributions, which indicated that the topics are actually doing the work. Then we used TF-IDF techniques to identify and extract the top content-words, and as the content words are extracted, the performance of the lexical model and the character n-grams dropped with respect to the size of words extracted. In other words, as the topics were extracted, the performance dropped; this phenomenon explained that the topics were doing the work.

5.2 Future Work

5.2.1 Spelling Errors

Reading the corpus also revealed that each language had spelling errors that are unique to each language. For example, Chinese writers tend to misspell *discuss* as *diskuss* and Bulgarian tend to misspell *discover* as *diskover*. Also, as discussed in Chapter 2, Koppel used spelling errors as a feature set in his experiment, and he learned that there was a relatively large number of incorrect usages of double consonants in the Spanish corpus [15]. Although we did not investigate the types of spelling errors in depth, we used the statistical measurements to learn how frequently writers misspelled words in general.

Spelling Errors								
	Bulgarian	Chinese	Czech	French	Japanese	Native	Russian	Spanish
Mean	3.81	6.595	7.82	5.19	4.045	7.1	5.13	11.885
Standard deviation	3.189	5.011	5.810	3.626	4.494	6.082	5.817	8.328

Table 5.1: Average and standard deviation of spelling errors for each language

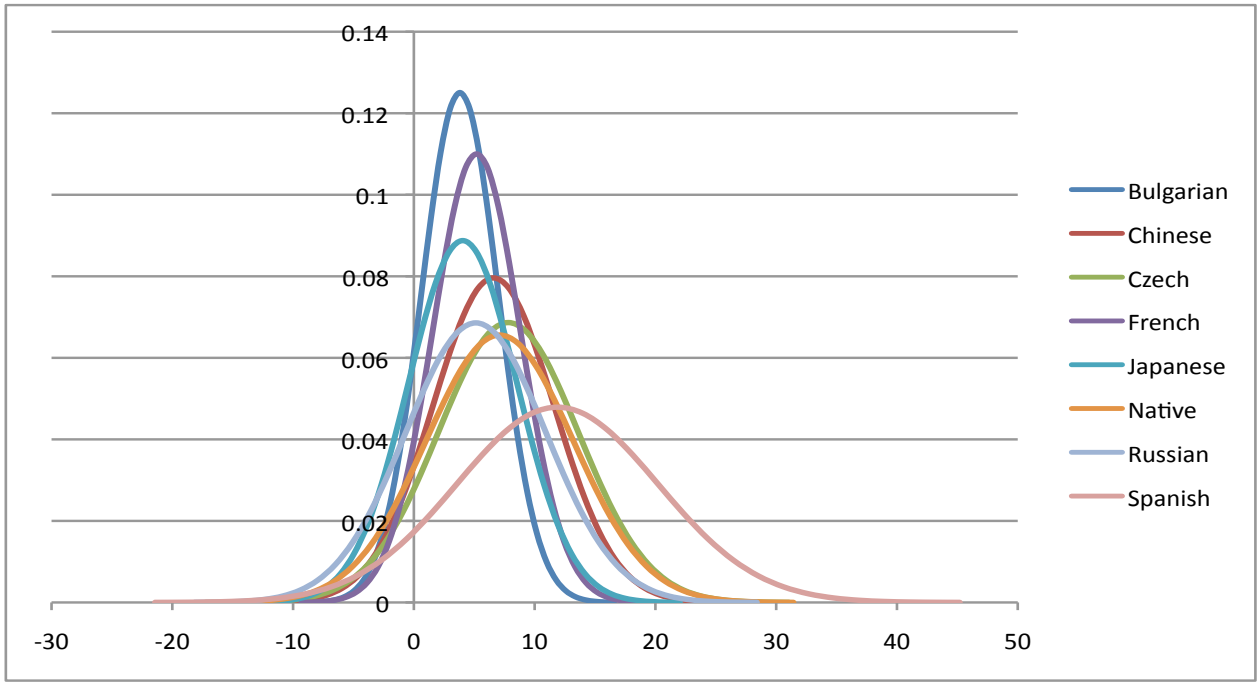


Figure 5.1: Bell curves (x-axis: number of misspelled words)

Table 5.1 presents the average number of spelling errors per document and the standard deviation for each language, and then we used the means and standard deviations to graph the normal

distributions as shown in Figure 5.1. The normal distribution shows that the Spanish distribution is more shifted to the right, and the peak is lower but wider, which indicates that Spanish writers make spelling errors more often than writers of other languages. Another interesting observation we made was that the mean value of spelling errors in the Native corpus is not significantly lower than the other languages. Consequently, when we pulled out the misspelled words in the Native corpus, most of the errors consisted of adding *s* to the non-countable nouns such as *Britains*, and slang or made up words that were not in the dictionary list, such as *Europhobic* and *Europeanism*. There are significant indications that using spelling errors as a feature set could improve the overall performance; therefore, it is worth investigating this method in future research. A simple way to test the role of spelling errors would be to find out whether removing misspelled words makes a difference.

5.2.2 Grammatical Errors

We discussed in Chapter 2 that L1-L2 language transfer influences the learners' learning of an L2, and that influence is a probably result of learners making grammatical mistakes such as misplaced modifiers, misused determiners, and errors in subject-verb agreement, just to name few. If there is a way to accurately capture these error patterns that is unique to a particular language, it will be an invaluable feature set for building an accurate model for each language, but such a task is very difficult. There are many off-the-shelf grammar-correction tools, but as far as we know, there is no tool that can capture these grammatical errors precisely enough to apply it to this type of problem. Some of the feature sets we used, such as POS n-grams and the distribution of transformation rules, could capture some grammatical patterns that are not usually found in writings by native speakers, but the Stanford parser, which we used for parsing sentences and POS tagging in this research, is created to work for sentences that are written grammatically correctly; consequently, although POS n-grams performed relatively well on the corpus written by native Chinese, it is difficult to measure how well POS n-grams and the distribution of transformation rules actually capture the effects of L1-L2 language transfer, especially grammatical errors influenced by L1-L2 language transfer. Wong and his team chose three grammatical error types (subject-verb agreement, noun-number disagreement, and misuse of determiners), and used them as their few feature sets using an in-house built tool that just captures these three grammatical error types; however, adding these grammatical error types did not improve the overall performance compared to the performance without them [16]. Also, the tool Wong and his team built produced 49% false positives, which may have affected their results. Wong concluded that either grammatical errors are not a good indicator or capturing

just three error types was not enough to produce any significant change in the result [16]. In future work, using more types of grammatical errors will be worth investigating.

Furthermore, as discussed in Chapter 2, different languages have different word order; for example, Japanese use Subject-Object-Verb order, and other languages have their own rules governing positions of adjectives and adverbs; these differences influence learners' writings in some way. If there is a particular grammatical error type that is made by writers of just one particular language, that will be a great indicator for modeling that language, and if there is such an error type, that is what we need to find. Using many different types of grammatical errors does not necessarily mean that they are good features, but we have to find those errors that help us to distinguish one language from the rest. Although there are linguistic theories about which error types may work better than others, conceived by studying the differences of the language's grammatical structure, the only way to confirm these hypotheses is to test and verify them. The challenging part of this task is to build a tool that can precisely capture these error types.

5.2.3 Chinese

The results show that Chinese outperformed the rest of the languages in all feature sets we used; Character n-grams' f-scores reached 0.9, function words reached 0.8, and the POS n-grams reached close to 0.8 while those of other languages (except Japanese) were around 0.5, and the word unigrams' f-scores reached 0.93. We also learned that topic words such as *Hong Kong* and *Cyber* were driving these high f-scores; however, even when topic words were removed, Chinese continued to outperform the other languages. With the bag of words as a feature set, when about 3000 topic words were removed, Chinese f-scores were about 0.65 while those of other languages were down in 0.4 range. With the character trigrams, Chinese f-scores stayed about 0.6 while those of other languages dropped below 0.4 when about 6000 topic words were removed. There must be some other indications that are driving these Chinese performances beside topic words. One way to investigate this phenomenon is to repeat the empirical analysis of character trigrams after topic words are extracted just as we did in Chapter 4 Section 4 to learn what drives the character-trigram's performance even after topic words are removed.

5.2.4 Noun Modifiers

After observing the corpus, we also noticed that some writers tend to stay with simple sentence styles (subject-verb-object), and they used sentences that had few adjectives as noun modifiers more often. Therefore, it maybe worth investigating the complexity of usages of noun modifiers according to language. If a learner's L1 uses a different grammatical structure from English to

describe a noun, the learner may try to avoid using complex noun modifiers such as prepositional phrases or noun phrases or participial phrases or infinitive phrases or adjective clauses as modifiers. Also, the order of modifiers in front of nouns may be challenging for some learners, so theyt may try to stay with simple one adjective-type modifiers. One way to test this hypothesis is to use the parsed trees of the Stanford parser. For example, Figure 5.2 shows a parsed tree of the sentence *I am a swimming champion in New York.* The first NP (noun phrase) captures *swimming champion in New York*, and inside of the first NP, there is another noun phrase (*a swimming champion*) followed by a prepositional phrase (PP) (*in New York*). From the parsed tree in Figure 5.2, we can learn that there is a prepositional phrase that modifies a noun, and there are two modifiers (*a* and *swimming*) that modify the same noun where *a* is one distance away from the describing noun and *swimming* is zero distance away from the describing noun. Therefore, in future research, it may be worth investigating whether using both the POS tags that modify nouns (including their distance from the describing noun) and the frequency of usage of phrases that modify nouns, such as preposition phrase of participial phrase, as a feature set would improve the overall performance.

```
I am a swimming champion in New York.
(ROOT
  (S
    (NP (PRP I ))
    (VP (VBP am )
     (NP
       (NP (DT a ) (VBB swimming) (NN champion ))
       (PP (IN in)
         (NP (NNP New) (NNP York)))))
    (. .)))
```

Figure 5.2: Stanford parser output

5.2.5 Phonological Transfer

As discussed in Chapter 2, Section 8, Rappoport conducted an in-depth investigation to learn what was driving the performance of character bigrams, and he concluded that they might be capturing language transfer effects at the level of basic sounds and short sound sequences [10]. We, on the other hand, concluded that character trigrams were simply simulating the lexical model through empirical analysis. However, when about 20% of topic words were extracted, while the bag of words' f-score dropped close to 0.3 for all languages, the character trigrams

and quadgrams performed significantly better than the bag of words. In the case of Chinese, as we have seen in the previous section, when about 20% of topic words were extracted, the character trigrams' f-score was up in the 0.6 range, while that of the bag of words was down in the 0.35 range. Character trigrams and quadgrams were tested after spaces were removed, cases were all converted to upper case, punctuation was removed, and the words were stemmed; consequently, we can conclude that character trigrams and quadgrams are capturing some other indications such as traits influenced by L1-L2 phonological transfer, as Rappoport concluded in his research. Therefore, it may be worth investigating whether L1-L2 phonological transfer influences the character n-grams' performances, and if that is the case, it will be interesting to find out if this feature set will improve the overall performance when it is combined with the other feature sets. One way to investigate the phonological transfer hypothesis is by building a character trigrams model from each group's L1 corpus and comparing that to the same models built from L2 corpora. For example, build a character trigrams model from a corpus written in Spanish and call this the Spanish L1 model. Then also build character trigrams models for Spanish L2, French L2, Czech L2, and Native L2. If we can demonstrate that the Spanish L1 model has a closer relationsip to the Spanish L2 model than it is to other languages' L2 models, we can conclude that character trigrams capture L1-L2 phonological transfer to some degree.

5.3 Concluding Remarks

The results of this research show that it is possible to differentiate authors' native languages based on their writing in English by exploring all their syntactic, lexical, and character styles, using models generated by Maximum Entropy. We have learned that the strongest indication of native provenance is in lexical usage when we observed that the bag of words performed better than the other feature sets, followed by character trigrams where these were shown to be simply simulating the word model. However, our result is not robust enough to apply in real world applications yet. As discussed above, there are many future avenues that can be investigated to improve the quality of this research, and the size of the corpus we used in this research may not be big enough to accurately capture the representation of each group's writing style.

APPENDIX A:
Confusion Matrices

	Classified As								
Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	13.7	0.7	0.4	1.2	2.0	0.5	0.6	0.9	20.0
Bulgarian	0.1	15.2	0.1	0.8	0.9	0.2	1.5	1.2	20.0
Chinese	0.9	0.4	16.4	0.5	0.3	0.7	0.5	0.3	20.0
Czech	0.7	1.4	0.4	13.3	0.6	0.3	3.0	0.3	20.0
French	0.7	1.3	0.1	0.9	13.5	0.1	1.2	2.2	20.0
Japanese	0.8	0.3	0.7	0.6	0.3	16.5	0.5	0.3	20.0
Russian	0.3	2.4	0.2	1.8	1.7	0.5	11.9	1.2	20.0
Spanish	0.4	1.0	0.1	0.7	2.1	0.1	1.7	13.9	20.0
Total	17.6	22.7	18.4	19.8	21.4	18.9	20.9	20.3	160.0

Table A.1: Character bigrams (MegaM)

	Classified As								
Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	16.5	0.4	0.1	0.7	0.7	0.5	0.1	1.0	20.0
Bulgarian	0.0	17.3	0.1	0.4	0.4	0.1	1.1	0.6	20.0
Chinese	0.4	0.4	17.8	0.4	0.1	0.3	0.4	0.2	20.0
Czech	0.4	1.2	0.0	15.4	0.2	0.4	1.9	0.5	20.0
French	0.4	1.0	0.0	0.9	15.4	0.0	0.9	1.4	20.0
Japanese	0.5	0.1	0.5	0.5	0.3	17.7	0.1	0.3	20.0
Russian	0.1	1.8	0.2	1.7	0.9	0.3	14.0	1.0	20.0
Spanish	0.1	0.3	0.0	0.4	1.9	0.0	1.2	16.1	20.0
Total	18.4	22.5	18.7	20.4	19.9	19.3	19.7	21.1	160.0

Table A.2: Character trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.8	0.5	0.2	1.0	0.8	0.4	0.5	0.8	20.0
Bulgarian	0.1	17.1	0.1	0.3	0.4	0.1	1.2	0.7	20.0
Chinese	0.6	0.3	17.7	0.4	0.1	0.3	0.4	0.2	20.0
Czech	0.4	0.9	0.0	14.7	0.6	0.5	2.4	0.5	20.0
French	0.3	1.2	0.1	0.7	15.3	0.1	1.0	1.3	20.0
Japanese	0.5	0.2	0.6	0.4	0.4	17.6	0.1	0.2	20.0
Russian	0.1	2.3	0.1	1.7	0.9	0.2	13.6	1.1	20.0
Spanish	0.3	0.6	0.0	0.4	2.3	0.0	1.5	14.9	20.0
Total	18.1	23.1	18.8	19.6	20.8	19.2	20.7	19.7	160.0

Table A.3: Character bigrams & trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	12.6	0.7	0.8	1.0	2.3	1.1	0.6	0.9	20.0
Bulgarian	0.6	12.7	0.6	2.3	0.9	0.2	2.0	0.7	20.0
Chinese	0.7	0.5	16.2	0.2	0.6	1.1	0.4	0.3	20.0
Czech	0.6	1.6	0.2	12.4	0.8	1.0	2.6	0.8	20.0
French	1.5	1.2	0.5	1.0	11.7	0.2	1.7	2.2	20.0
Japanese	1.3	0.2	1.0	0.4	0.5	15.0	1.1	0.5	20.0
Russian	0.7	2.1	0.5	2.5	1.3	0.4	11.1	1.4	20.0
Spanish	1.3	0.9	0.3	1.1	2.3	0.2	1.3	12.6	20.0
Total	19.3	19.9	20.1	20.9	20.4	19.2	20.8	19.4	160.0

Table A.4: Function words (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	12.6	0.7	0.8	1.0	2.3	1.1	0.6	0.9	20.0
Bulgarian	0.6	12.7	0.6	2.3	0.9	0.2	2.0	0.7	20.0
Chinese	0.7	0.5	16.2	0.2	0.6	1.1	0.4	0.3	20.0
Czech	0.6	1.6	0.2	12.4	0.8	1.0	2.6	0.8	20.0
French	1.5	1.2	0.5	1.0	11.7	0.2	1.7	2.2	20.0
Japanese	1.3	0.2	1.0	0.4	0.5	15.0	1.1	0.5	20.0
Russian	0.7	2.1	0.5	2.5	1.3	0.4	11.1	1.4	20.0
Spanish	1.3	0.9	0.3	1.1	2.3	0.2	1.3	12.6	20.0
Total	19.3	19.9	20.1	20.9	20.4	19.2	20.8	19.4	160.0

Table A.5: Top 200 POS bigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	8.5	1.0	1.9	1.6	1.8	1.5	1.7	2.0	20.0
Bulgarian	1.5	8.4	0.4	2.0	2.2	0.6	2.8	2.1	20.0
Chinese	1.7	0.7	15.0	0.2	0.2	1.0	0.8	0.4	20.0
Czech	1.4	2.3	0.2	7.6	1.0	1.5	4.6	1.4	20.0
French	1.3	2.7	0.5	1.4	8.2	0.5	1.6	3.8	20.0
Japanese	1.6	0.4	1.7	2.4	0.4	11.0	1.6	0.9	20.0
Russian	2.3	2.5	0.3	3.3	1.9	1.7	5.9	2.1	20.0
Spanish	1.8	2.1	0.3	0.9	3.5	1.1	1.8	8.5	20.0
Total	20.1	20.1	20.3	19.4	19.2	18.9	20.8	21.2	160.0

Table A.6: Top 200 POS trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	9.8	1.3	1.5	0.9	2.8	0.8	1.3	1.6	20.0
Bulgarian	0.7	9.7	0.1	1.9	2.6	0.2	2.7	2.1	20.0
Chinese	1.0	0.4	16.4	0.4	0.3	0.7	0.5	0.3	20.0
Czech	1.4	2.7	0.4	8.6	1.3	1.3	3.1	1.2	20.0
French	1.0	2.3	0.5	1.1	10.3	0.3	1.7	2.8	20.0
Japanese	1.4	0.5	0.7	1.3	0.7	14.0	1.2	0.2	20.0
Russian	1.3	2.5	0.2	4.1	1.6	0.8	8.3	1.2	20.0
Spanish	1.5	1.6	0.0	1.2	2.4	0.4	2.4	10.5	20.0
Total	18.1	21.0	19.8	19.5	22.0	18.5	21.2	19.9	160.0

Table A.7: Top 200 POS bigrams & trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.8	0.4	0.1	1.1	1.0	0.3	0.7	0.6	20.0
Bulgarian	0.1	17.1	0.1	0.3	0.4	0.1	1.2	0.7	20.0
Chinese	0.5	0.3	17.8	0.4	0.1	0.3	0.5	0.1	20.0
Czech	0.3	0.9	0.0	14.8	0.6	0.5	2.4	0.5	20.0
French	0.3	1.1	0.1	0.7	15.6	0.1	1.0	1.1	20.0
Japanese	0.5	0.2	0.6	0.4	0.4	17.6	0.1	0.2	20.0
Russian	0.0	2.0	0.2	1.8	1.1	0.2	13.6	1.1	20.0
Spanish	0.3	0.6	0.0	0.4	2.1	0.0	1.4	15.2	20.0
Total	17.8	22.6	18.9	19.9	21.3	19.1	20.9	19.5	160.0

Table A.8: Function words and character bigrams & trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.8	0.4	0.0	1.1	1.0	0.4	0.3	1.0	20.0
Bulgarian	0.1	16.9	0.1	0.3	0.5	0.1	1.3	0.7	20.0
Chinese	0.6	0.3	17.7	0.5	0.1	0.3	0.4	0.1	20.0
Czech	0.4	1.2	0.1	14.8	0.8	0.3	2.0	0.4	20.0
French	0.4	1.0	0.0	0.5	15.9	0.0	1.1	1.1	20.0
Japanese	0.5	0.2	0.7	0.5	0.3	17.3	0.3	0.2	20.0
Russian	0.1	1.5	0.1	1.8	0.8	0.3	14.2	1.2	20.0
Spanish	0.2	0.9	0.0	0.5	1.7	0.0	1.1	15.6	20.0
Total	18.1	22.4	18.7	20.0	21.1	18.7	20.7	20.3	160.0

Table A.9: Function words and character bigrams & trigrams and top 200 POS bigrams & trigrams (MegaM)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	10.4	1.6	0.6	1.8	4.2	0.2	0.5	0.7	20.0
Bulgarian	0.0	17.6	0.0	1.0	0.3	0.2	0.8	0.1	20.0
Chinese	0.9	0.8	15.2	1.1	0.1	0.4	1.2	0.3	20.0
Czech	1.3	2.9	0.1	10.9	1.6	0.3	2.6	0.3	20.0
French	0.5	2.9	0.0	2.3	8.4	0.2	2.8	2.9	20.0
Japanese	0.9	0.4	0.7	1.2	0.5	14.7	1.4	0.2	20.0
Russian	0.3	4.4	0.0	2.8	1.0	0.1	11.2	0.2	20.0
Spanish	0.6	4.6	0.0	1.2	2.4	0.0	2.4	8.8	20.0
Total	14.9	35.2	16.6	22.3	18.5	16.1	22.9	13.5	160.0

Table A.10: Character bigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	8.3	1.6	1.0	1.0	4.6	1.0	1.5	1.0	20.0
Bulgarian	0.1	17.7	0.0	0.9	0.6	0.2	0.3	0.2	20.0
Chinese	0.4	0.6	15.2	1.1	0.3	0.4	1.1	0.9	20.0
Czech	0.6	4.2	0.0	8.8	2.0	1.0	2.4	1.0	20.0
French	0.0	3.3	0.2	2.8	8.5	0.1	1.6	3.5	20.0
Japanese	0.5	0.6	0.7	1.5	0.5	14.4	1.0	0.8	20.0
Russian	0.4	4.9	0.0	4.0	1.0	0.3	8.1	1.3	20.0
Spanish	0.3	5.3	0.0	2.0	2.4	0.4	2.5	7.1	20.0
Total	10.6	38.2	17.1	22.1	19.9	17.8	18.5	15.8	160.0

Table A.11: Character trigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.8	0.5	0.2	1.0	0.8	0.4	0.5	0.8	20.0
Bulgarian	0.1	17.1	0.1	0.3	0.4	0.1	1.2	0.7	20.0
Chinese	0.6	0.3	17.7	0.4	0.1	0.3	0.4	0.2	20.0
Czech	0.4	0.9	0.0	14.7	0.6	0.5	2.4	0.5	20.0
French	0.3	1.2	0.1	0.7	15.3	0.1	1.0	1.3	20.0
Japanese	0.5	0.2	0.6	0.4	0.4	17.6	0.1	0.2	20.0
Russian	0.1	2.3	0.1	1.7	0.9	0.2	13.6	1.1	20.0
Spanish	0.3	0.6	0.0	0.4	2.3	0.0	1.5	14.9	20.0
Total	18.1	23.1	18.8	19.6	20.8	19.2	20.7	19.7	160.0

Table A.12: Character bigrams & trigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	13.2	1.2	0.9	0.8	1.7	1.1	0.4	0.7	20.0
Bulgarian	0.4	15.3	0.4	1.4	0.7	0.4	0.9	0.5	20.0
Chinese	0.6	1.0	17.2	0.0	0.3	0.3	0.3	0.3	20.0
Czech	2.3	2.5	0.5	9.8	1.6	1.0	1.9	0.4	20.0
French	1.6	3.1	0.2	1.6	10.0	0.0	1.3	2.2	20.0
Japanese	1.3	0.6	1.8	1.0	0.5	13.6	0.5	0.7	20.0
Russian	1.7	3.6	0.6	2.7	1.0	0.4	9.2	0.8	20.0
Spanish	1.2	2.8	0.3	0.8	2.3	0.6	0.6	11.4	20.0
Total	22.3	30.1	21.9	18.1	18.1	17.4	15.1	17.0	160.0

Table A.13: Function words (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.9	2.2	0.0	1.2	0.0	0.6	0.1	0.0	20.0
Bulgarian	5.7	11.2	0.1	2.5	0.0	0.5	0.0	0.0	20.0
Chinese	9.9	2.6	4.0	1.6	0.0	1.9	0.0	0.0	20.0
Czech	7.9	3.2	0.0	7.1	0.0	1.7	0.1	0.0	20.0
French	12.0	4.1	0.0	2.4	0.1	1.3	0.0	0.1	20.0
Japanese	4.9	2.8	0.0	3.2	0.0	9.0	0.1	0.0	20.0
Russian	7.0	4.8	0.2	5.9	0.1	1.5	0.4	0.1	20.0
Spanish	10.0	4.5	0.0	3.8	0.1	0.7	0.0	0.9	20.0
Total	73.3	35.4	4.3	27.7	0.3	17.2	0.7	1.1	160.0

Table A.14: Top 200 POS bigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	10.6	0.6	1.1	1.1	0.6	3.4	2.6	0.0	20.0
Bulgarian	1.6	8.3	0.6	2.0	0.0	3.2	4.0	0.3	20.0
Chinese	4.2	1.2	10.5	0.2	0.0	2.6	1.3	0.0	20.0
Czech	2.9	1.4	0.4	3.8	0.0	8.0	3.2	0.3	20.0
French	6.1	3.7	1.1	1.4	1.0	3.8	2.5	0.4	20.0
Japanese	1.9	1.1	0.4	0.9	0.0	14.7	0.9	0.1	20.0
Russian	2.5	2.6	0.9	3.5	0.1	5.6	4.7	0.1	20.0
Spanish	5.3	3.7	0.6	1.8	0.5	4.1	3.0	1.0	20.0
Total	35.1	22.6	15.6	14.7	2.2	45.4	22.2	2.2	160.0

Table A.15: Top 200 POS trigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	15.6	1.6	0.1	1.1	0.2	0.6	0.8	0.0	20.0
Bulgarian	4.1	11.2	0.2	2.7	0.1	0.6	1.1	0.0	20.0
Chinese	7.3	1.6	8.8	0.4	0.0	1.4	0.5	0.0	20.0
Czech	6.4	2.6	0.0	6.1	0.0	3.4	1.5	0.0	20.0
French	11.1	3.7	0.1	1.3	0.5	1.5	1.2	0.6	20.0
Japanese	4.4	1.9	0.2	1.7	0.0	11.2	0.6	0.0	20.0
Russian	6.1	3.8	0.3	4.6	0.1	2.1	2.7	0.3	20.0
Spanish	9.5	4.0	0.0	2.3	0.2	1.0	1.4	1.6	20.0
Total	64.5	30.4	9.7	20.2	1.1	21.8	9.8	2.5	160.0

Table A.16: Top 200 POS bigrams & trigrams (Naive Bayes)

Classified As

Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	10.0	1.2	0.8	1.7	4.6	0.4	0.7	0.6	20.0
Bulgarian	0.1	17.7	0.0	1.0	0.5	0.2	0.3	0.2	20.0
Chinese	0.9	0.7	15.2	1.1	0.0	0.4	0.9	0.8	20.0
Czech	0.9	3.5	0.0	10.6	1.6	0.4	1.8	1.2	20.0
French	0.3	2.8	0.1	3.0	8.9	0.1	1.4	3.4	20.0
Japanese	0.9	0.6	0.5	1.4	0.5	14.4	1.0	0.7	20.0
Russian	0.4	5.3	0.0	4.1	0.9	0.2	7.9	1.2	20.0
Spanish	0.2	5.0	0.0	2.2	2.2	0.4	2.0	8.0	20.0
Total	13.7	36.8	16.6	25.1	19.2	16.5	16.0	16.1	160.0

Table A.17: Function words and character bigrams & trigrams (Naive Bayes)

	Classified As								
Actual	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish	Total
Native	14.7	2.1	0.0	0.7	1.7	0.5	0.3	0.0	20.0
Bulgarian	1.0	10.0	0.0	3.1	0.7	0.0	5.2	0.0	20.0
Chinese	2.2	0.3	14.4	0.5	0.1	0.6	1.9	0.0	20.0
Czech	2.6	1.4	0.0	13.0	0.5	0.0	2.5	0.0	20.0
French	1.5	2.2	0.0	1.6	10.1	0.0	4.5	0.1	20.0
Japanese	2.2	0.7	0.0	0.3	0.0	16.3	0.5	0.0	20.0
Russian	1.7	2.0	0.0	0.9	0.4	0.3	14.7	0.0	20.0
Spanish	3.2	2.4	0.0	1.8	2.6	0.1	5.7	4.2	20.0
Total	29.1	21.1	14.4	21.9	16.1	17.8	35.3	4.3	160.0

Table A.18: Function words and character bigrams & trigrams and top 200 POS bigrams & trigrams (Naive Bayes)

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B:
MegaM Vs. Naive Bayes

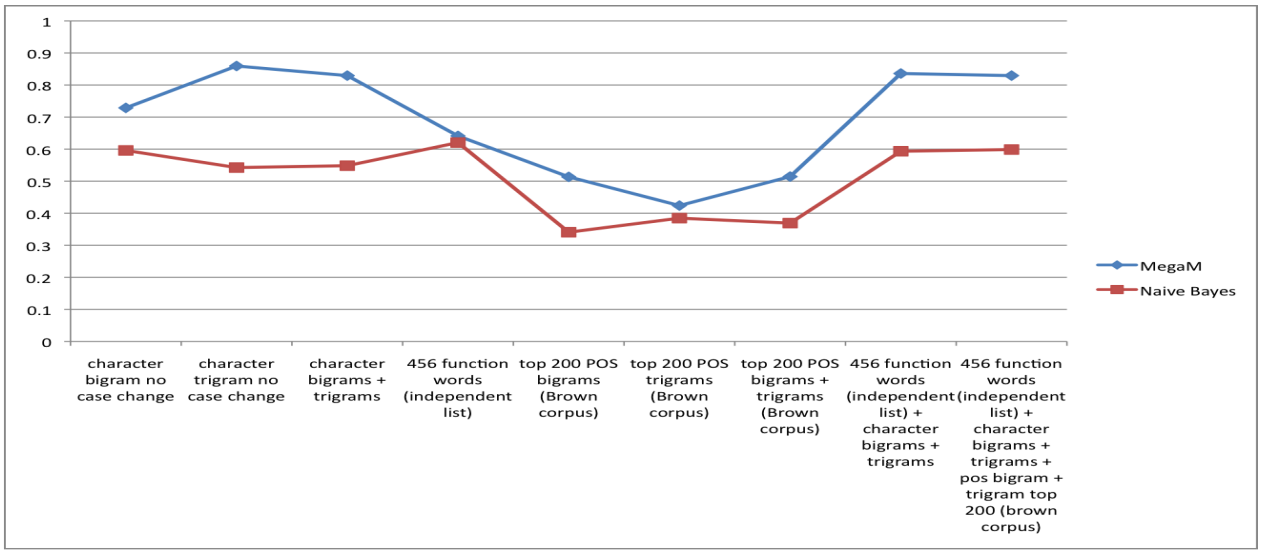


Figure B.1: MegaM vs. Naive Bayes (Native f-scores)

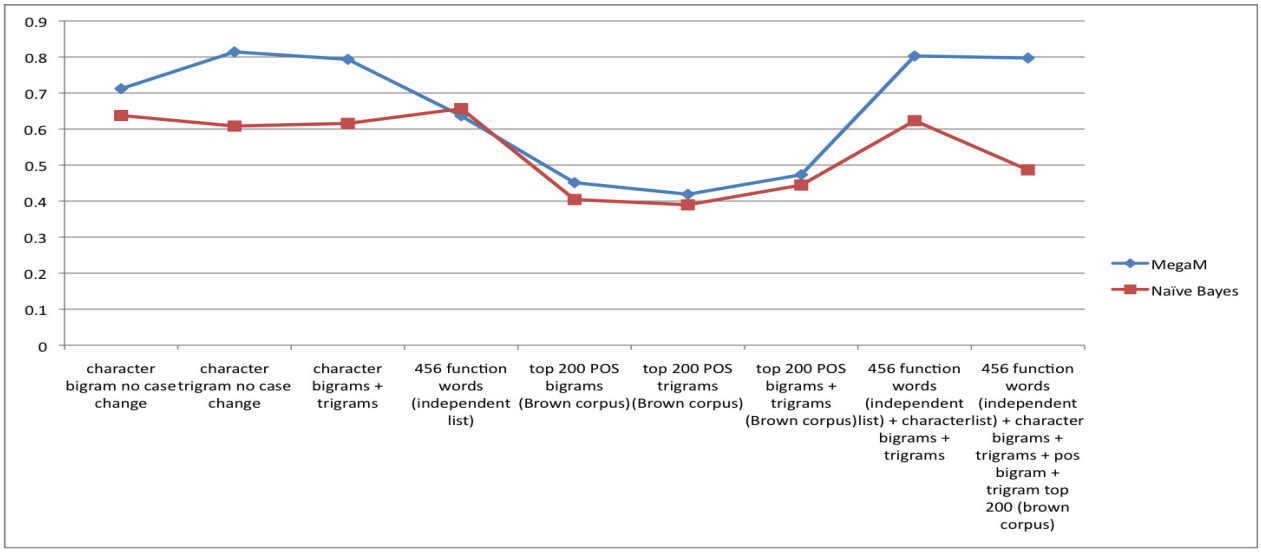


Figure B.2: MegaM vs. Naive Bayes (Bulgarian f-scores)

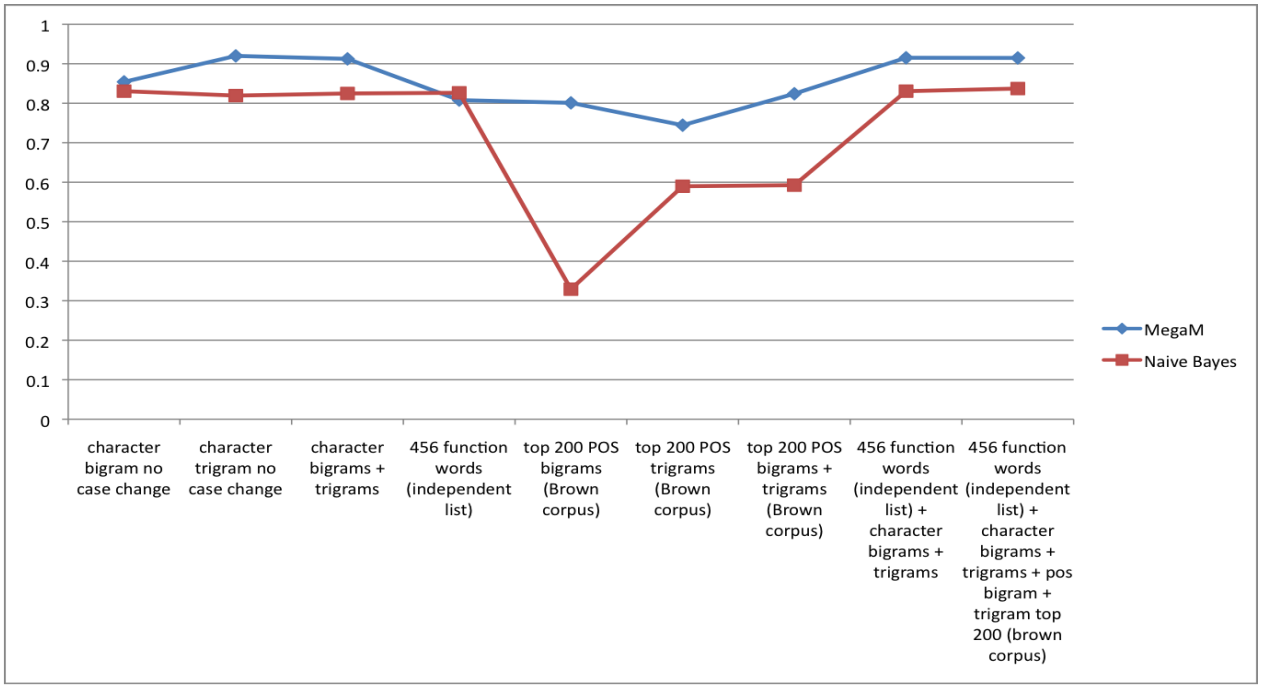


Figure B.3: MegaM vs. Naive Bayes (Chinese f-scores)

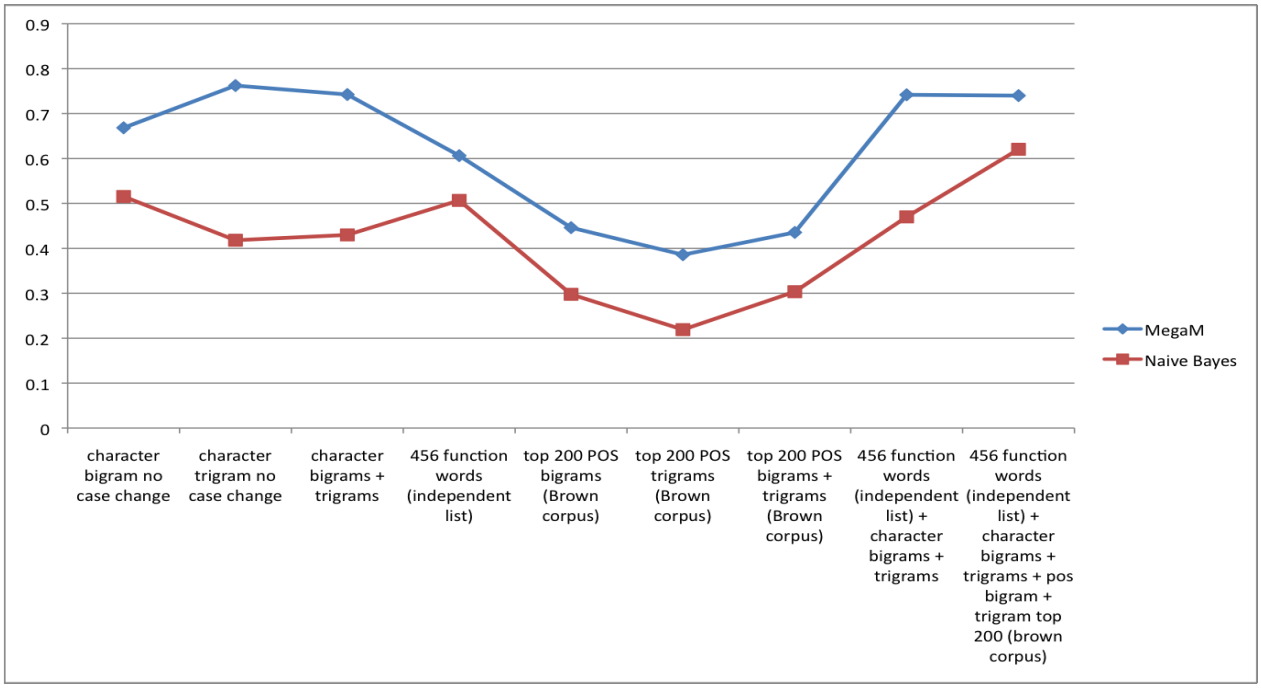


Figure B.4: MegaM vs. Naive Bayes (Czech f-scores)

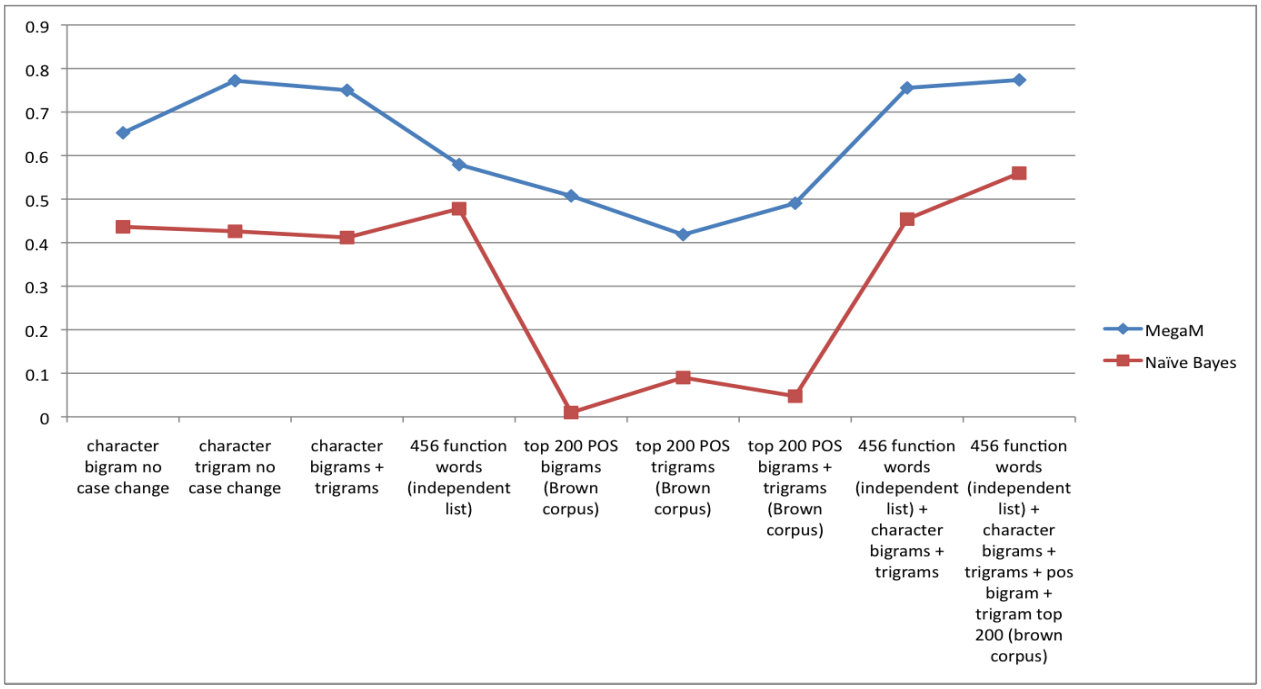


Figure B.5: MegaM vs. Naive Bayes (French f-scores)

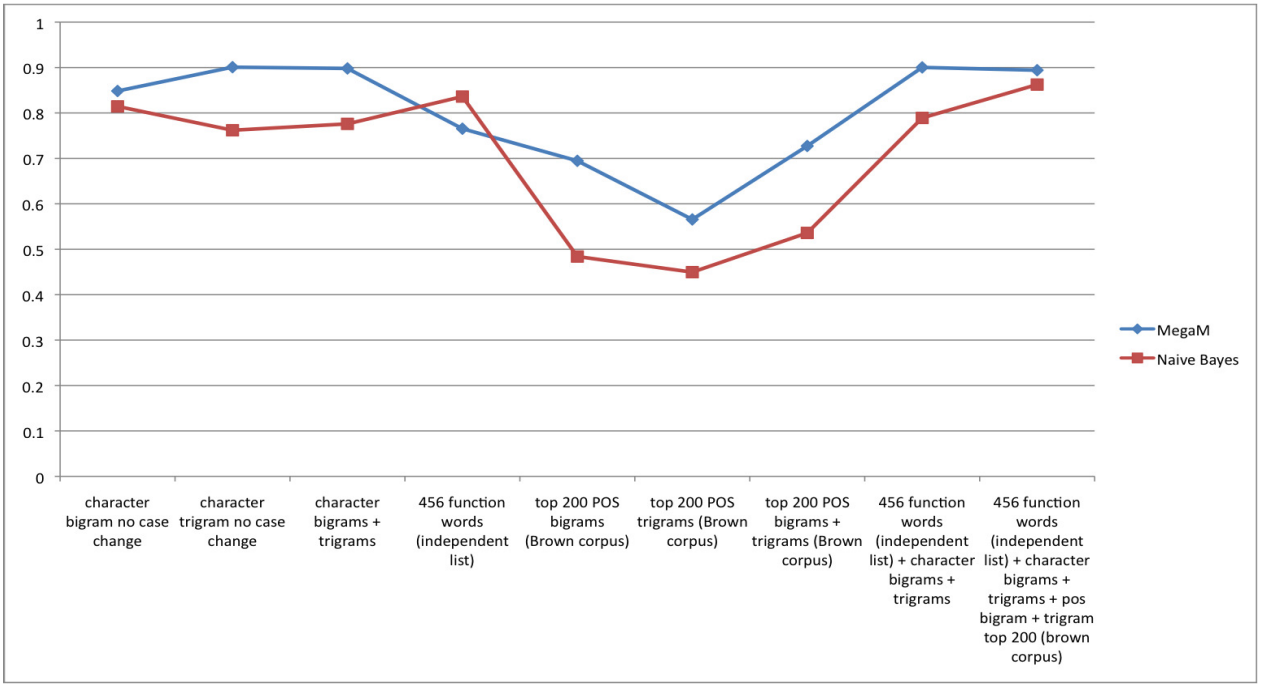


Figure B.6: MegaM vs. Naive Bayes (Japanese f-scores)

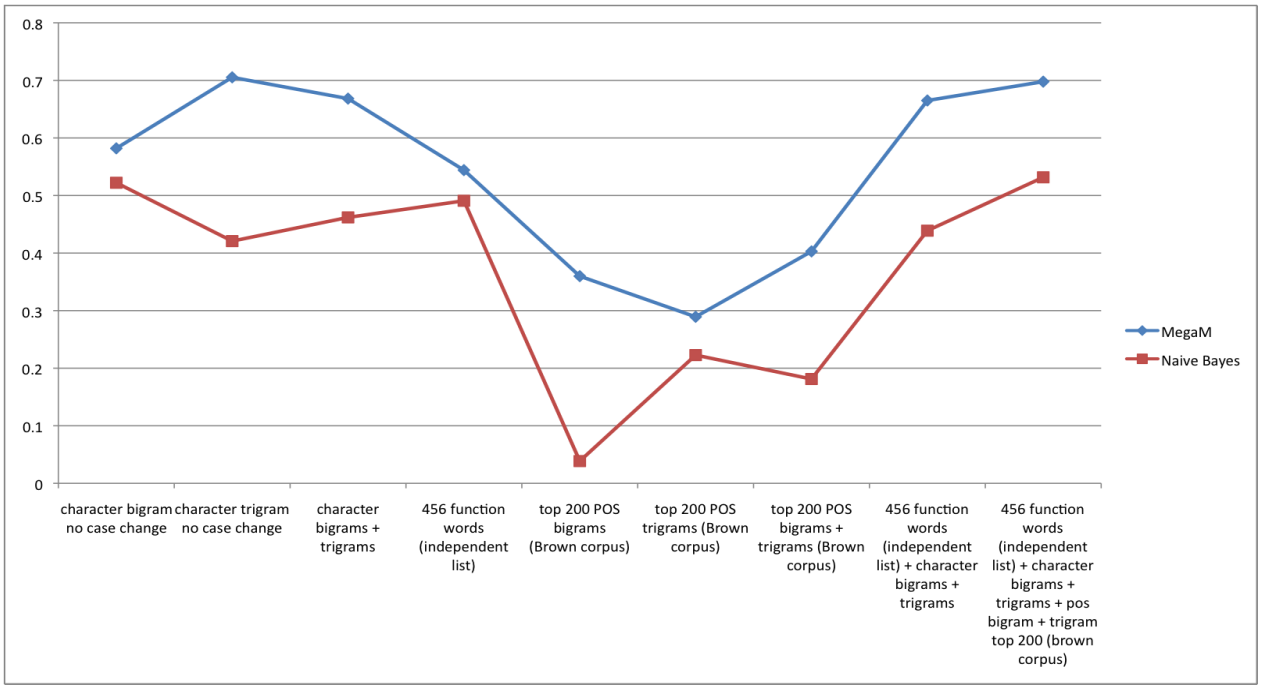


Figure B.7: MegaM vs. Naive Bayes (Russian f-scores)

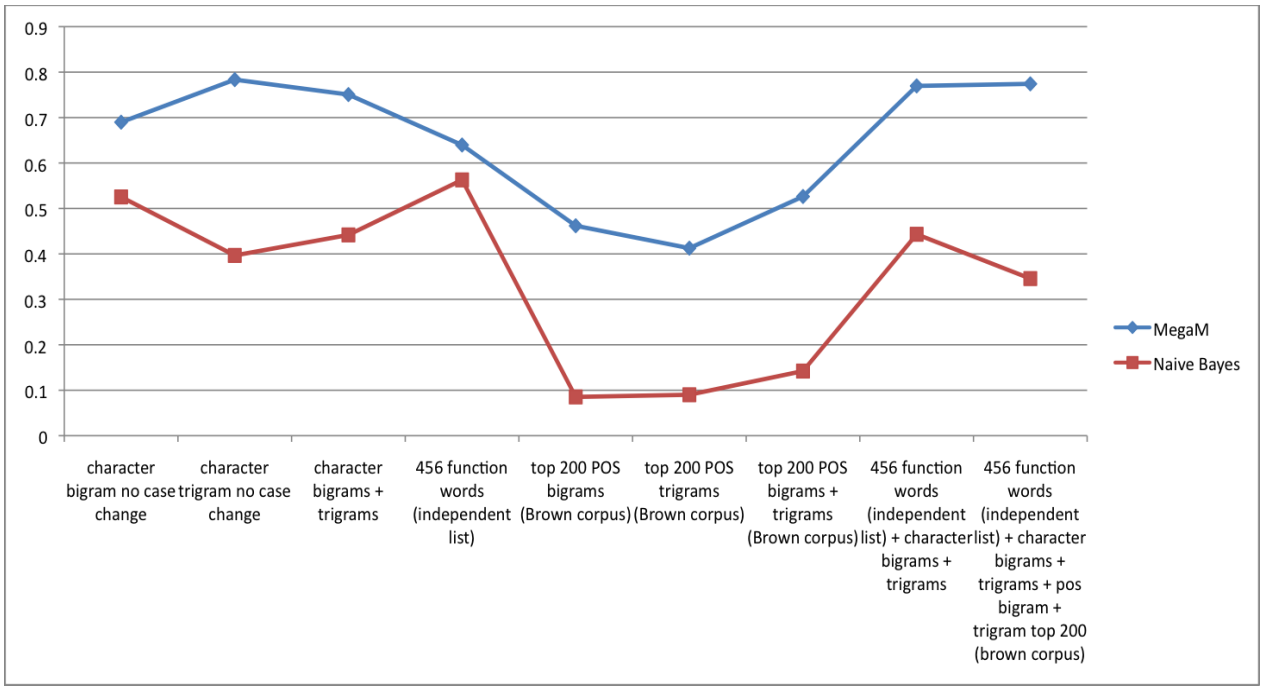


Figure B.8: MegaM vs. Naive Bayes (Spanish f-scores)

APPENDIX C:
Distribution of Transformation Rules

Machine Learning Tool: Naive Bayes									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
DTR	0.335 0.226	0.285	0.534	0.299	0.274	0.368	0.213	0.422	

CB	Character bigrams
CT	Character trigrams
CBT	Character bigrams & trigrams
FW	Function words
POSB	Top 200 POS bigrams
POST	Top 200 POS trigrams
POSBT	Top 200 POS bigrams & trigrams
FW CBT	Function words and character bigrams & trigrams
DTR	Distribution of transformation rules

Table C.1: Distribution of transformation rules

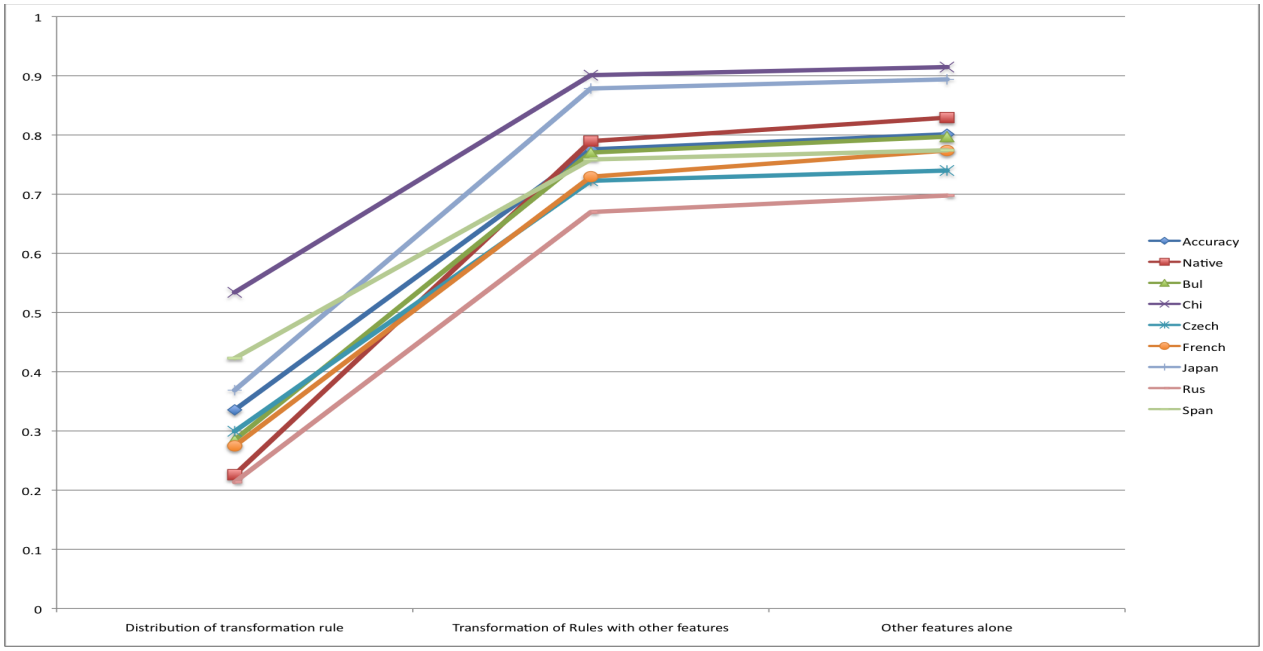


Figure C.1: Distribution of transformation rules and it's contribution

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D:
Intra-Regional Classification

Machine Learning Tool: Megam					
Features	Accuracy	Asian f-score	Slovic f-score	Romance f-score	Native
CB	0.75	*0.821*	0.730	0.697	0.754
CT	**0.841**	***0.885***	**0.808**	**0.816**	**0.857**
CBT	0.821	*0.877*	0.800	0.778	0.831
FW	0.715	*0.778*	0.705	0.666	0.710
POSB	0.597	*0.733*	0.549	0.490	0.618
POST	0.575	*0.665*	0.5481	0.535	0.551
POSBT	0.617	*0.755*	0.546	0.543	0.627
FW CBT	0.823	*0.875*	0.803	0.786	0.833
FW CBT POSBT	0.825	*0.876*	0.795	0.793	0.839

Table D.1: Accuracies and F-scores for Intra-regional classification

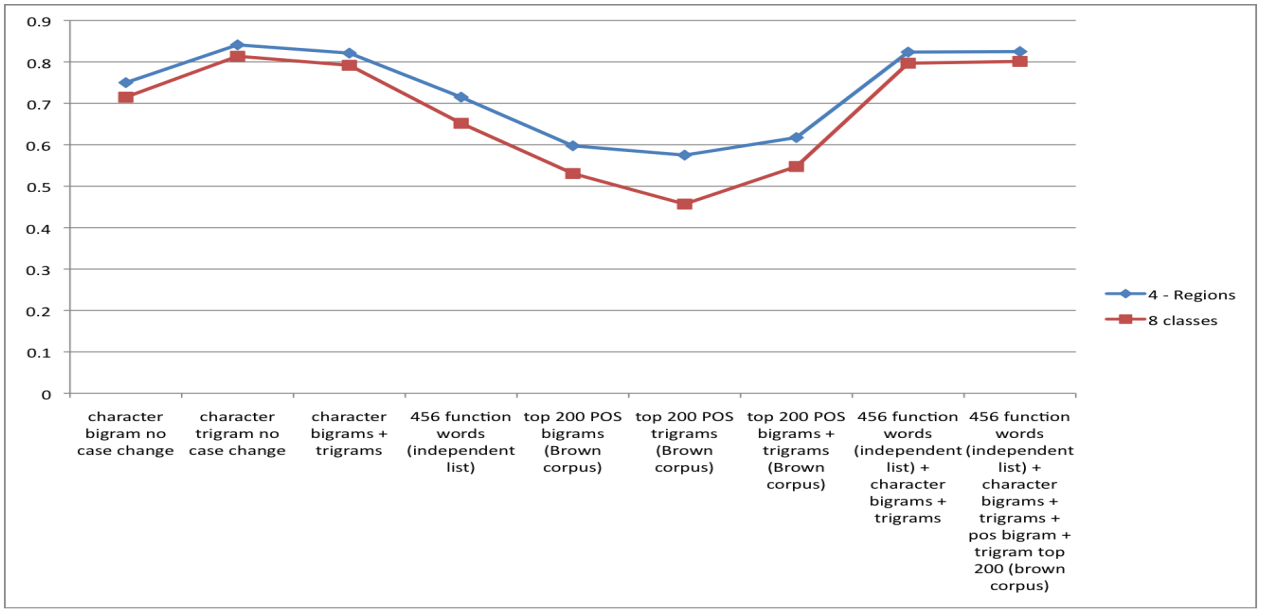


Figure D.1: Accuracies comparison between individual classification and intra-regional classification

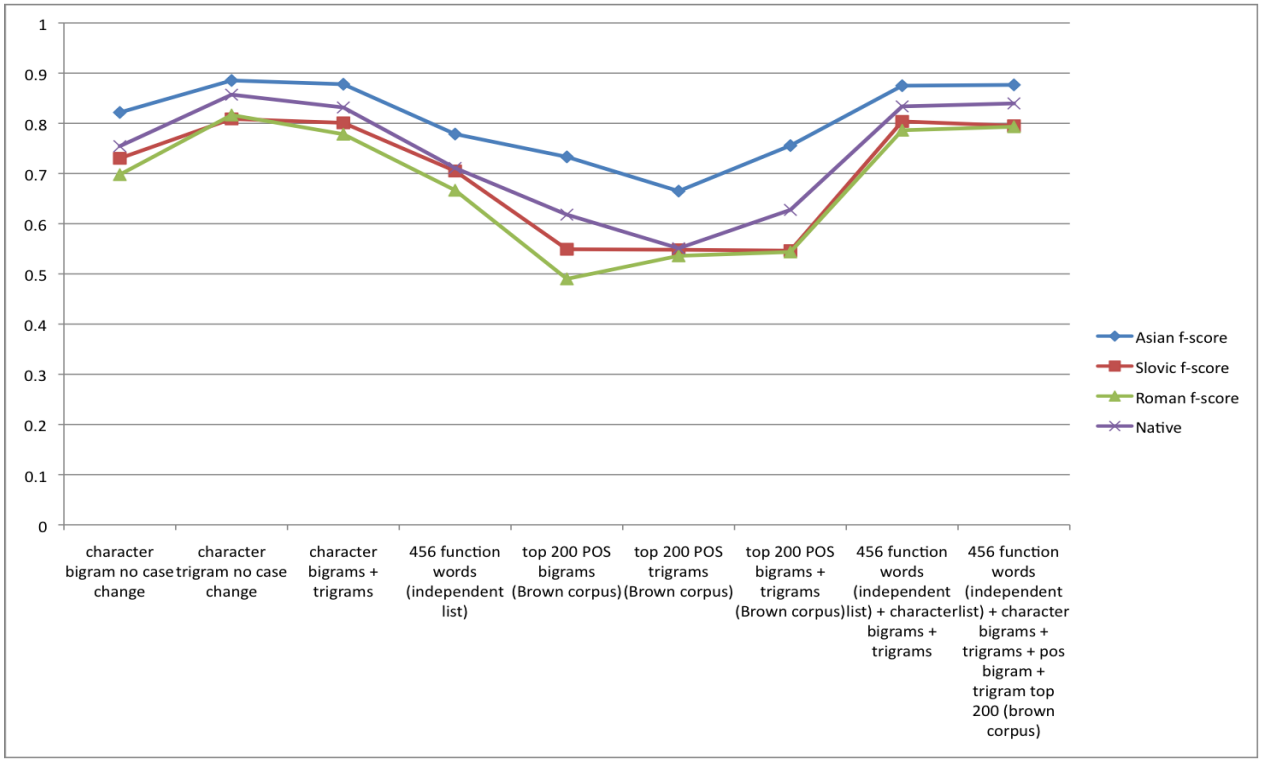


Figure D.2: Regions vs. each other

Machine Learning Tool: Megam			
Features	Accuracy	Chinese f-score	Japanese f-score
CB	0.932	0.931	*0.933*
CT	*0.945*	0.944	*0.945*
CBT	*0.950*	0.949	*0.950*
FW	*0.905*	0.904	*0.905*
POSB	0.902	0.902	0.902
POST	0.850	0.849	0.850
POSBT	0.910	*0.911*	0.908
FW CBT	0.952	0.951	*0.953*
FW CBT POSBT	***0.955***	**0.954**	***0.955***

Table D.2: Chinese vs. Japanese

Machine Learning Tool: Megam				
Features	Accuracy	Bulgarian f-score	Czech f-score	Russian f-score
CB	0.728	*0.798*	0.720	0.663
CT	**0.823**	***0.871***	**0.809**	**0.785**
CBT	0.8	*0.846*	0.787	0.763
FW	0.708	0.708	*0.732*	0.683
POSB	0.578	*0.658*	0.5481	0.525
POST	0.518	*0.582*	0.516	0.454
POSBT	0.583	*0.678*	0.546	0.525
FW CBT	0.810	*0.850*	0.803	0.773
FW CBT POSBT	0.806	*0.851*	0.803	0.762

Table D.3: Bulgarian vs. Czech vs. Russian

Machine Learning Tool: Megam			
Features	Accuracy	French f-score	Spanish f-score
CB	0.807	0.807	0.807
CT	**0.870**	**0.870**	**0.869**
CBT	0.835	0.835	0.834
FW	0.772	0.776	*0.768*
POSB	0.752	0.749	*0.755*
POST	0.682	0.668	*0.695*
POSBT	0.750	*0.751*	0.748
FW CBT	0.845	0.844	0.845
FW CBT POSBT	0.850	*0.851*	0.848

Table D.4: French vs. Spanish

Machine Learning Tool: MegaM									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CB	0.699	0.754	0.582	*0.766*	0.525	0.563	0.765	0.484	0.563
CT	**0.794**	***0.857***	**0.704**	0.836	**0.654**	**0.710**	**0.836**	**0.635**	**0.709**
CBT	0.780	0.831	0.677	*0.834*	0.630	0.650	0.833	0.611	0.649
FW	0.647	*0.710*	0.499	0.705	0.516	0.517	0.704	0.482	0.512
POSB	0.539	0.618	0.361	*0.661*	0.300	0.367	*0.661*	0.288	0.370
POST	0.488	0.551	0.319	*0.565*	0.283	0.358	0.564	0.249	0.372
POSBT	0.561	0.627	0.370	0.686	0.298	0.408	*0.688*	0.286	0.407
FW CBT	0.784	0.833	0.683	*0.834*	0.645	0.663	0.832	0.622	0.664
FW CBT POSBT	0.787	*0.839*	0.677	**0.837**	0.638	0.675	**0.836**	0.606	0.673

Table D.5: Accuracies and F-scores after pipelining

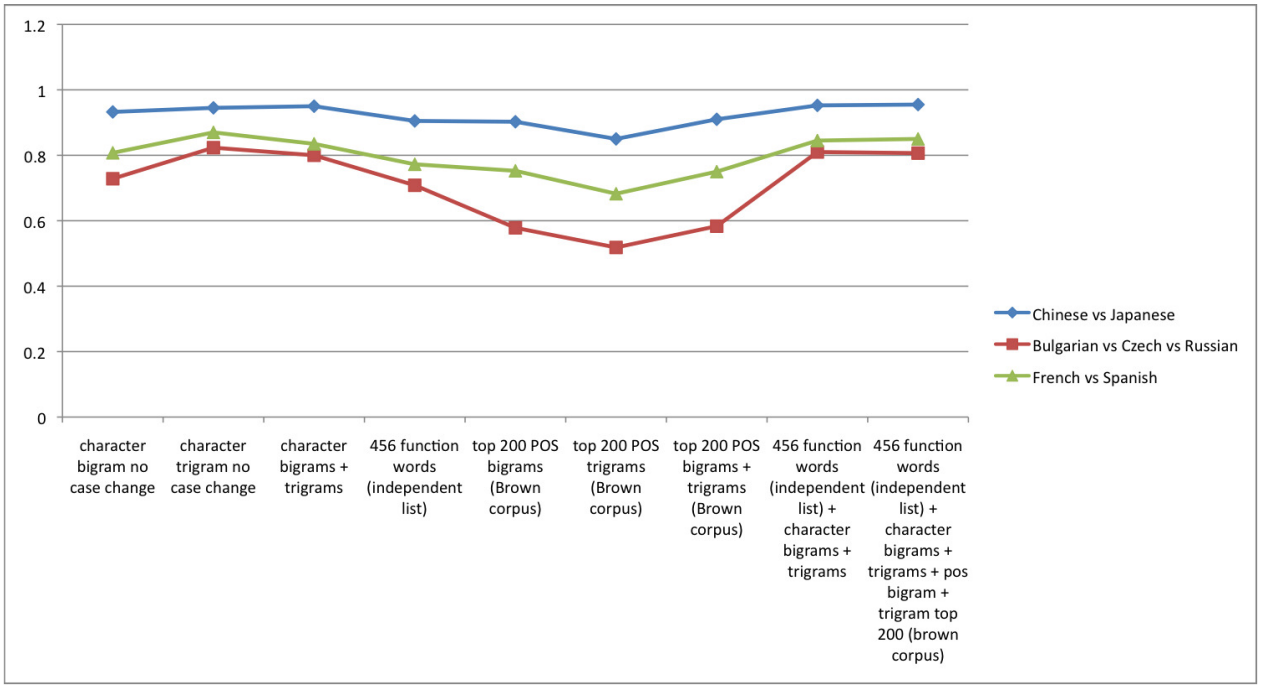


Figure D.3: Intran-regions (Accuracies)

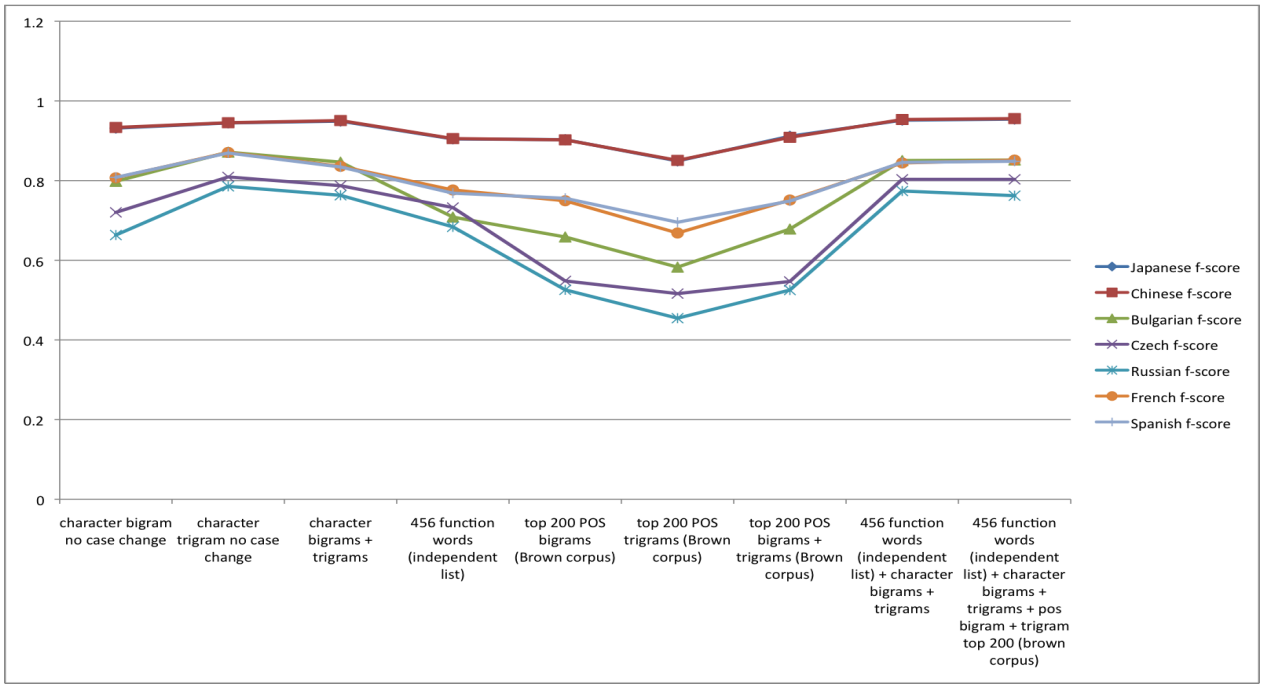


Figure D.4: Intra-regions (F-scores)

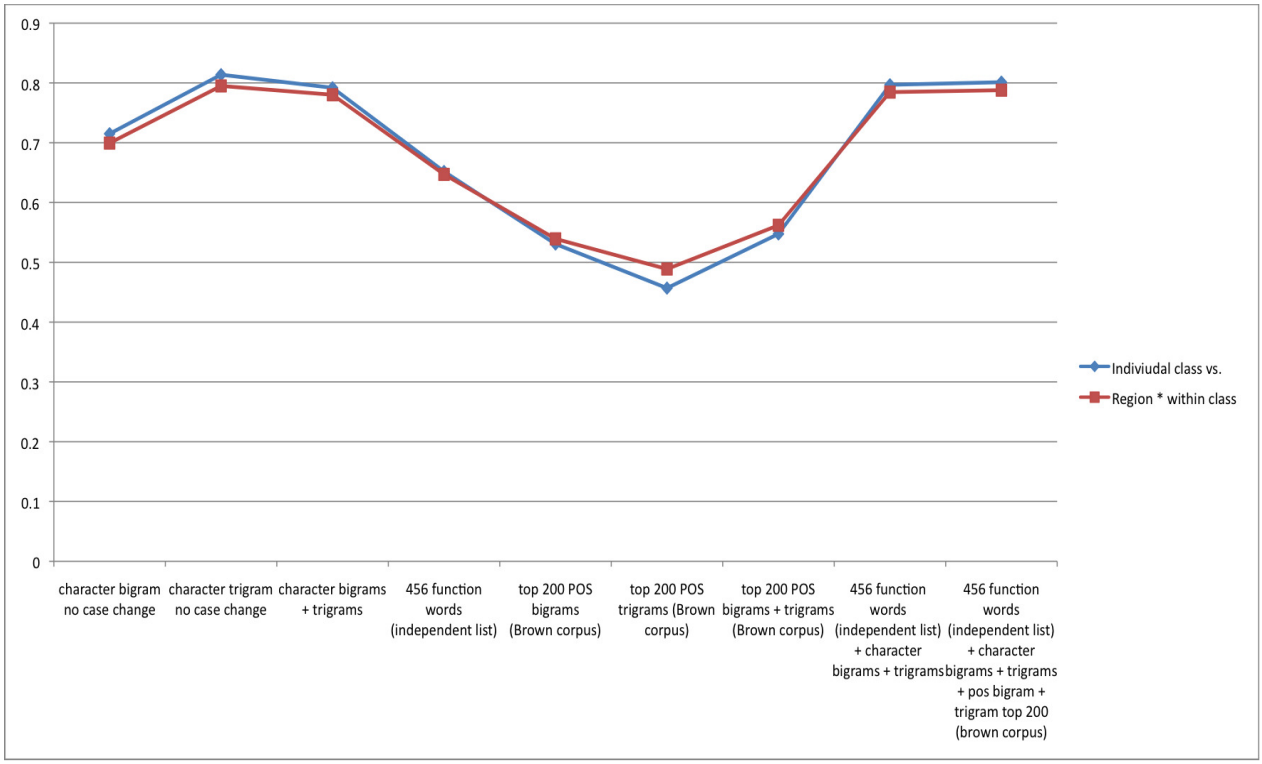


Figure D.5: Individual vs. Intra-regional piped value (Accuracies)

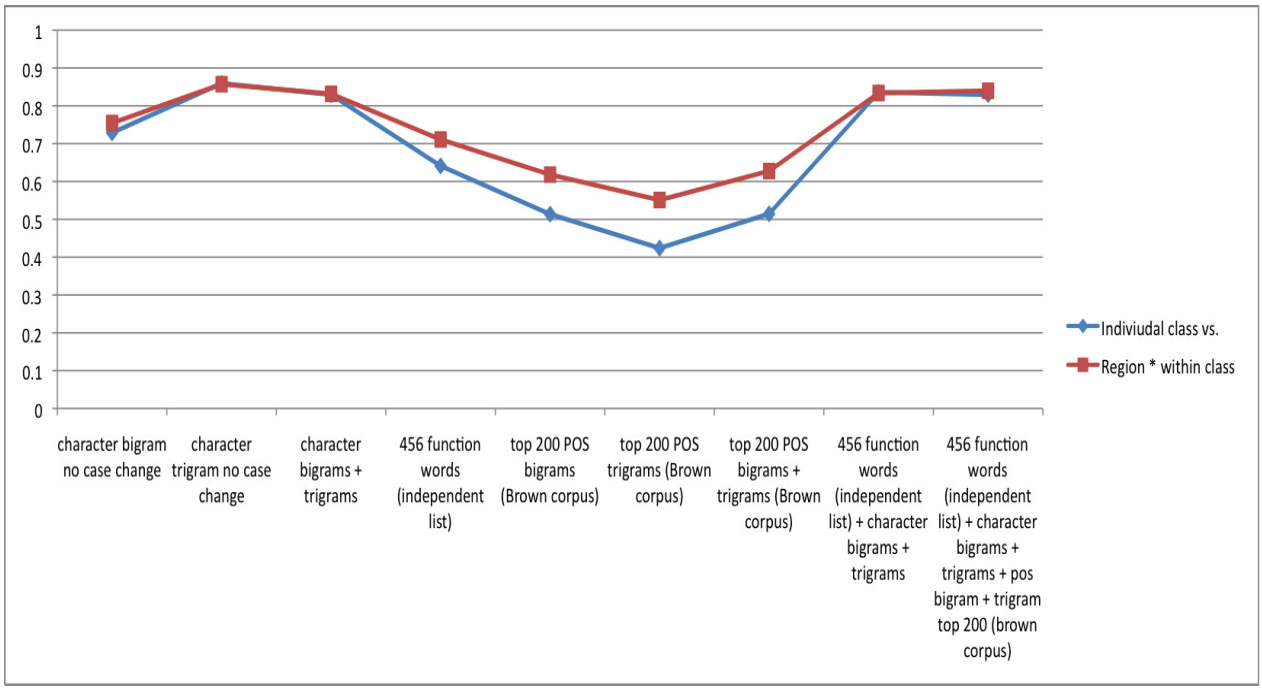


Figure D.6: Individual vs. Intra-regional piped value (Native F-scores)

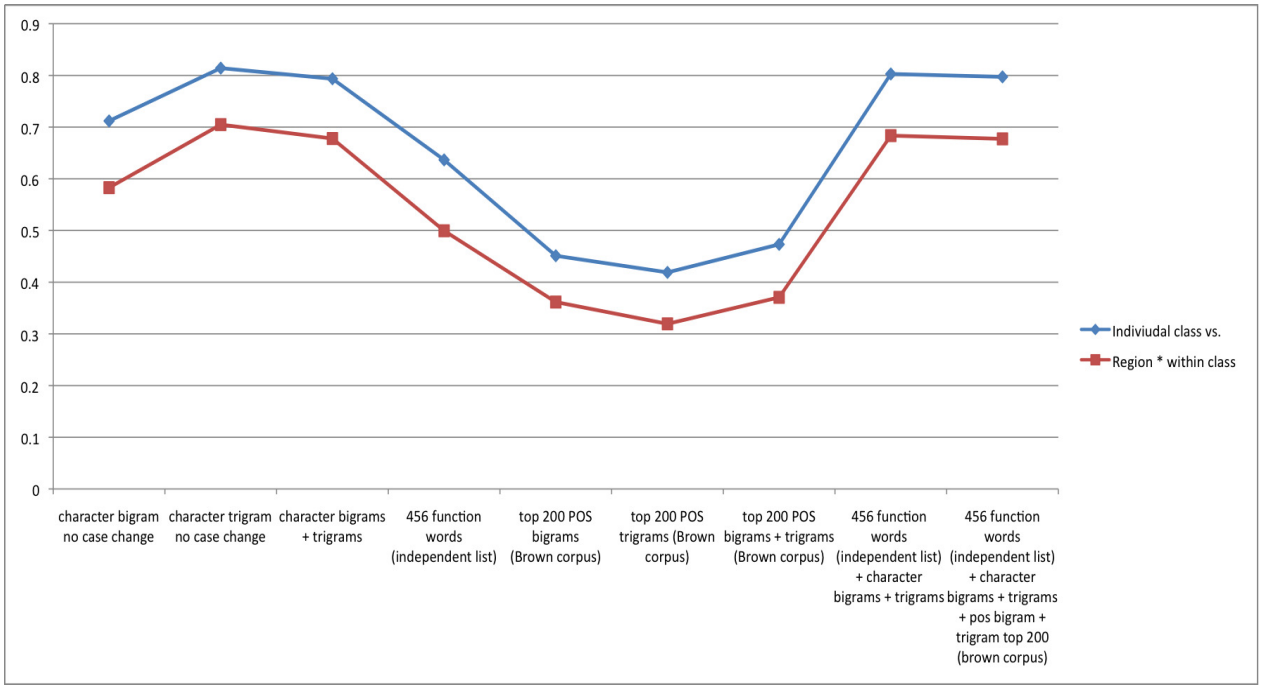


Figure D.7: Individual vs. Intra-regional piped value (Bulgarian F-scores)

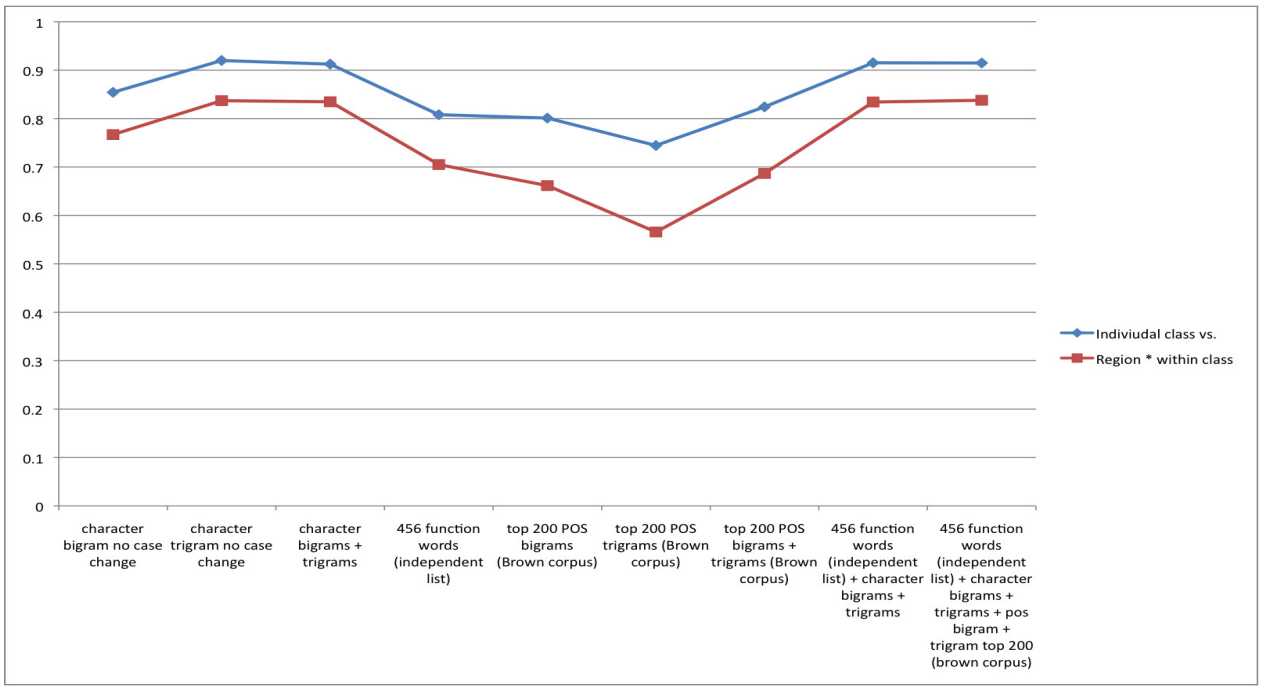


Figure D.8: Individual vs. Intra-regional piped value (Chinese F-scores)

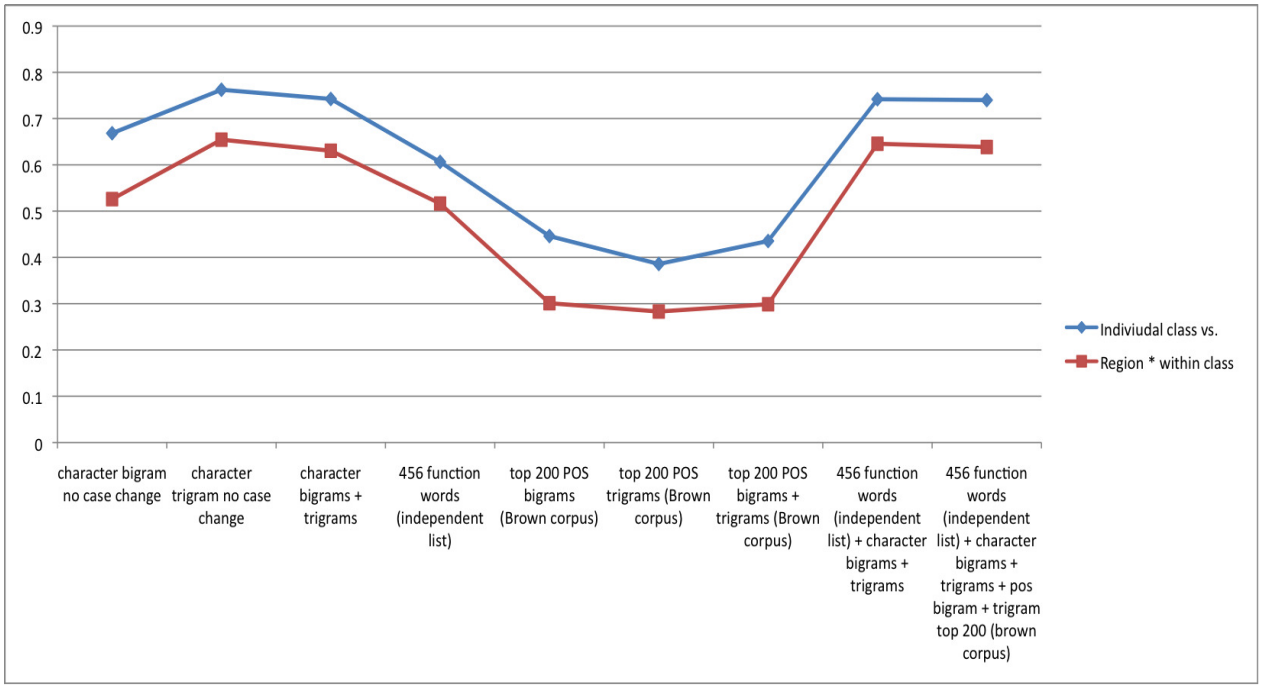


Figure D.9: Individual vs. Intra-regional piped value (Czech F-scores)

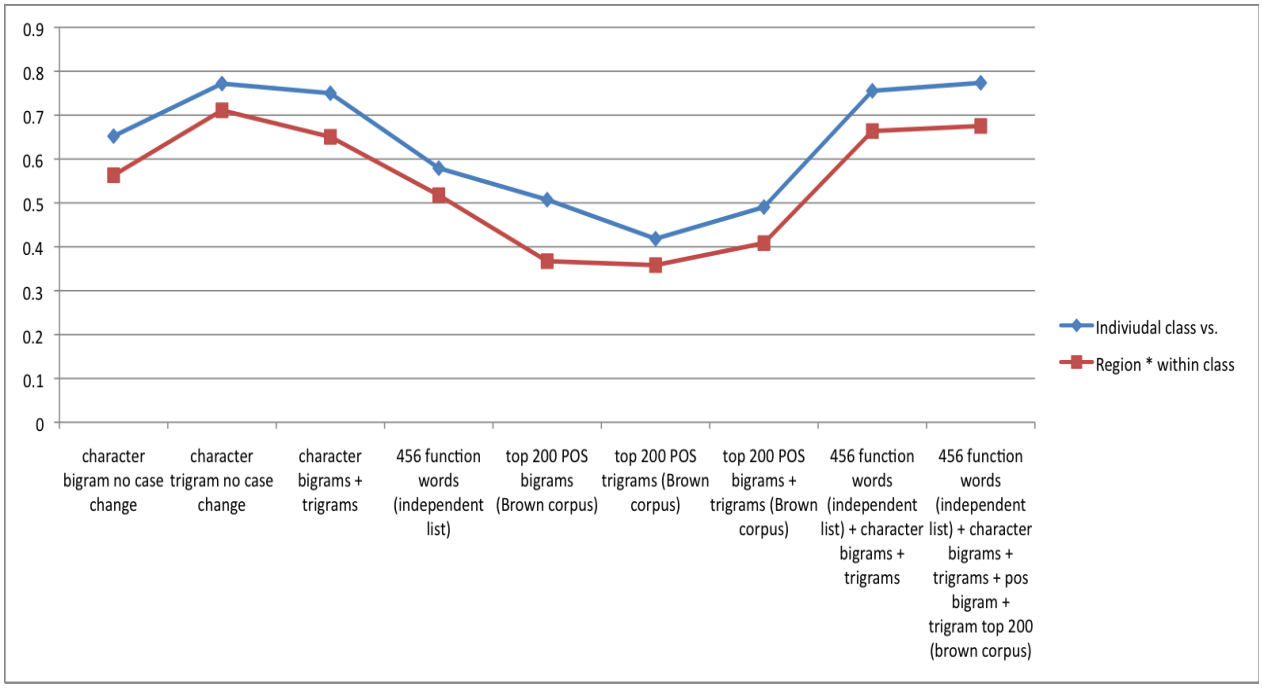


Figure D.10: Individual vs. Intra-regional piped value (French F-scores)

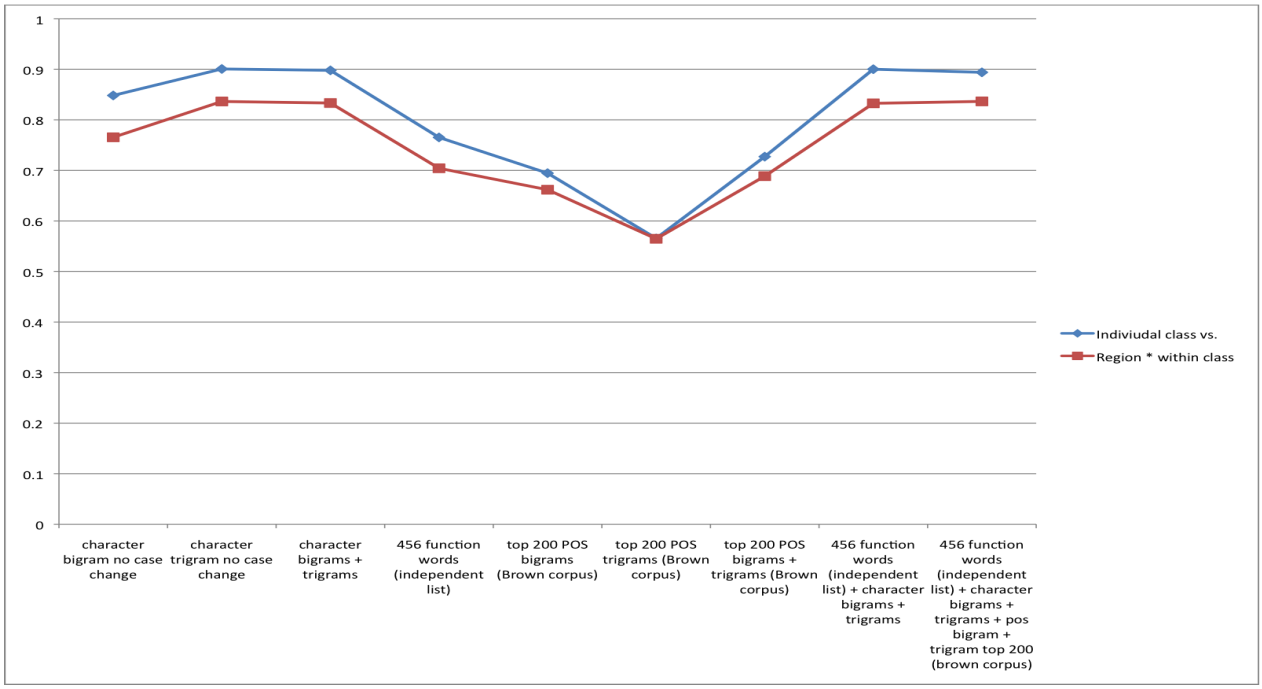


Figure D.11: Individual vs. Intra-regional piped value (Japanese F-scores)

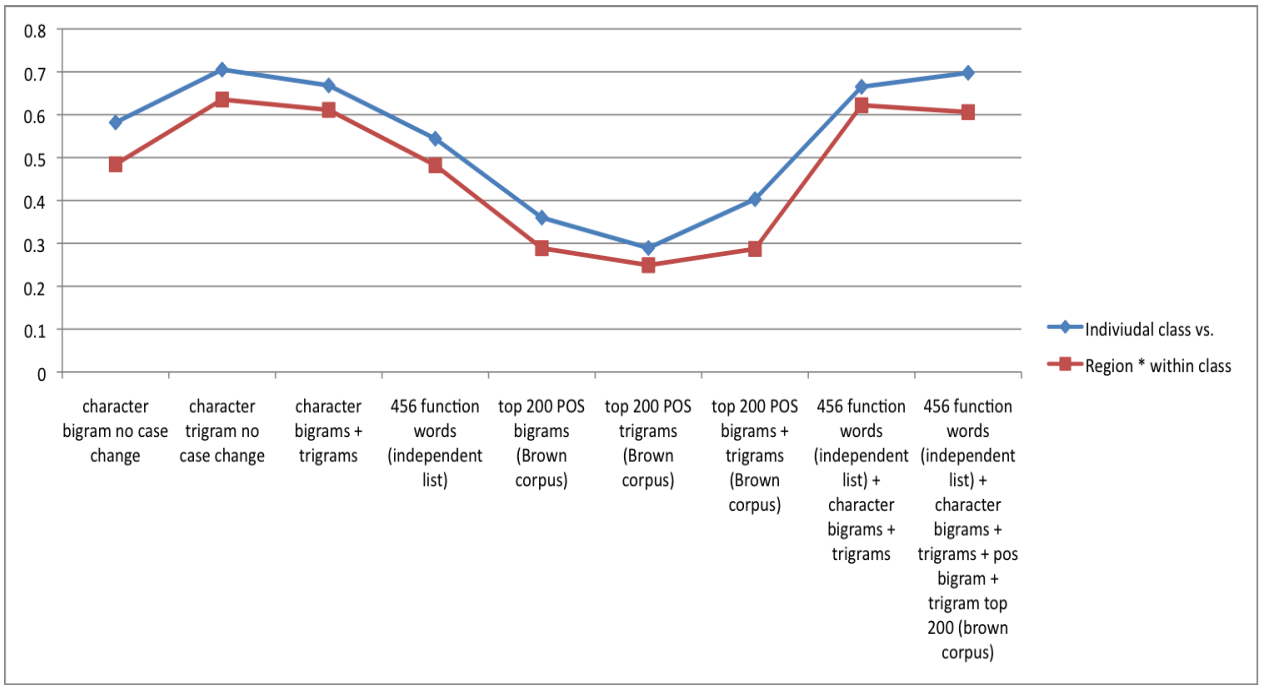


Figure D.12: Individual vs. Intra-regional piped value (Russian F-scores)

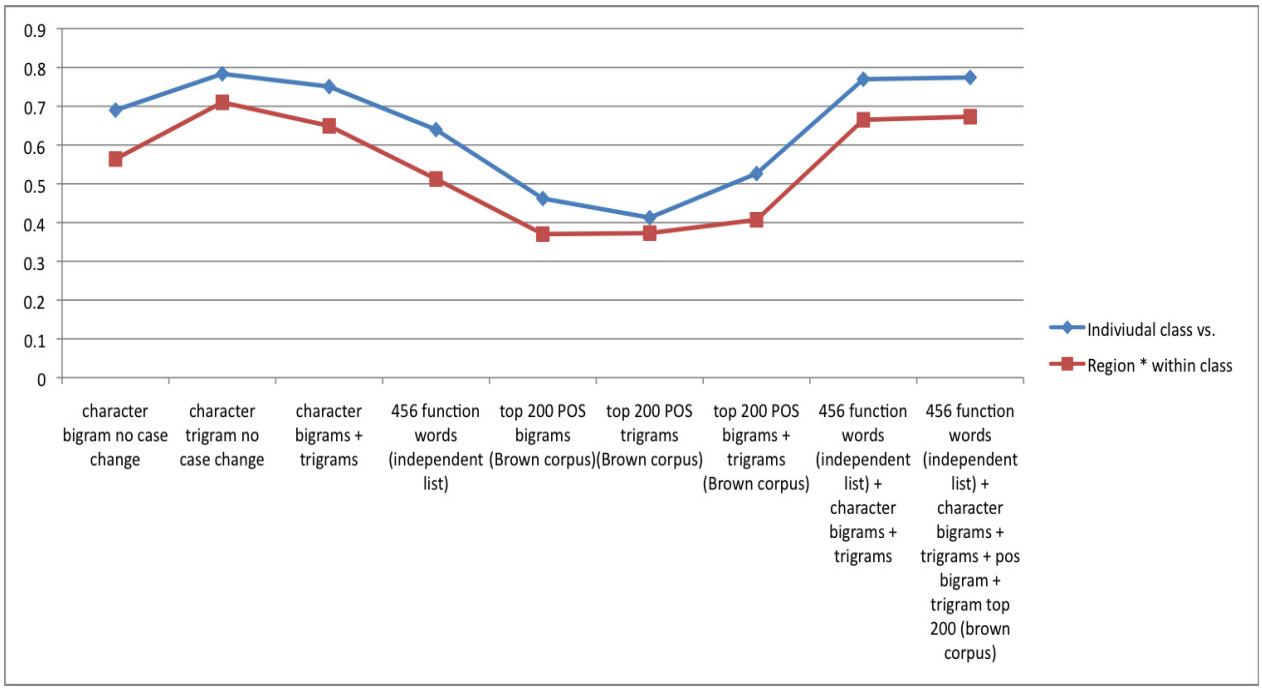


Figure D.13: Individual vs. Intra-regional piped value (Spanish F-scores)

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E:
Performances After Topics are Controlled

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
LDA	**0.561**	**0.610**	**0.675**	***0.823***	0.371	**0.567**	**0.762**	**0.308**	**0.346**
LDA 100	0.478	0.488	0.559	*0.753*	**0.443**	0.352	0.625	0.249	0.320
LDA 90	0.452	0.368	0.537	*0.710*	0.422	0.398	0.592	0.2166	0.339
LDA 80	0.471	0.432	0.492	*0.732*	0.427	0.491	0.577	0.281	0.352
LDA 70	0.390	0.229	0.537	*0.586*	0.381	0.301	0.472	0.295	0.284
LDA 60	0.382	0.374	0.457	*0.683*	0.188	0.345	0.457	0.206	0.287
LDA 50	0.352	0.285	0.338	*0.569*	0.311	0.322	0.450	0.248	0.220
LDA 40	0.318	0.232	0.281	*0.504*	0.274	0.407	0.433	0.162	0.195
LDA 30	0.278	0.286	0.205	*0.413*	0.240	0.288	0.390	0.177	0.108
LDA 25	0.226	0.200	0.225	0.281	0.183	0.147	*0.349*	0.170	0.195
LDA 20	0.236	0.238	0.176	*0.388*	0.268	0.130	0.313	0.141	0.130
LDA 15	0.206	0.171	0.143	*0.317*	0.148	0.171	0.318	0.164	0.140
LDA 10	0.226	0.265	0.207	0.296	0.181	0.205	*0.319*	0.143	0.152

LDA 100	LDA coefficients after extracting 245 words identified by the TF-IDF threshold 100

Table E.1: LDA coefficients for 50 topics as a feature set

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
BOW	**0.814**	**0.838**	**0.804**	***0.934***	**0.741**	**0.775**	**0.911**	0.705	**0.810**
BOW 100	0.782	0.786	0.799	*0.893*	0.730	0.731	0.819	0.693	0.805
BOW 90	0.789	0.807	0.779	*0.895*	0.740	0.751	0.830	**0.717**	0.797
BOW 80	0.774	0.818	0.765	*0.876*	0.732	0.742	0.808	0.688	0.769
BOW 70	0.751	0.764	0.746	*0.842*	0.723	0.716	0.797	0.656	0.761
BOW 60	0.736	0.727	0.764	*0.845*	0.701	0.696	0.7533	0.666	0.731
BOW 50	0.706	0.653	0.758	*0.824*	0.690	0.686	0.689	0.631	0.702
BOW 40	0.680	0.653	0.708	*0.830*	0.675	0.661	0.659	0.580	0.663
BOW 30	0.599	0.538	0.673	*0.773*	0.594	0.570	0.576	0.506	0.556
BOW 25	0.531	0.474	0.589	*0.705*	0.502	0.544	0.511	0.441	0.491
BOW 20	0.481	0.415	0.467	*0.648*	0.477	0.473	0.480	0.431	0.457
BOW 15	0.388	0.301	0.380	*0.491*	0.428	0.372	0.412	0.359	0.359
BOW 10	0.305	0.225	0.233	0.343	0.342	0.293	*0.370*	0.311	0.285

BOW 100	Bag of words after extracting 245 words identified by the TF-IDF threshold 100

Table E.2: Bag of words

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CT	**0.781**	**0.796**	**0.787**	***0.923***	**0.747**	**0.724**	**0.873**	**0.683**	**0.726**
CT 100	0.746	0.743	0.772	*0.88*	0.691	0.711	0.787	0.652	0.74
CT 90	0.716	0.688	0.743	*0.856*	0.695	0.672	0.757	0.615	0.704
CT 80	0.713	0.713	0.741	*0.878*	0.655	0.68	0.738	0.605	0.698
CT 70	0.688	0.613	0.703	*0.875*	0.643	0.69	0.738	0.555	0.683
CT 60	0.674	0.663	0.705	*0.842*	0.627	0.639	0.728	0.557	0.644
CT 50	0.641	0.605	0.676	*0.815*	0.583	0.588	0.707	0.542	0.612
CT 40	0.589	0.531	0.587	*0.788*	0.509	0.576	0.651	0.477	0.59
CT 30	0.529	0.461	0.576	*0.752*	0.449	0.491	0.577	0.409	0.519
CT 25	0.476	0.449	0.48	*0.733*	0.383	0.417	0.528	0.352	0.466
CT 20	0.437	0.392	0.459	*0.689*	0.358	0.351	0.497	0.294	0.43
CT 15	0.413	0.39	0.438	*0.648*	0.362	0.322	0.462	0.269	0.404
CT 10	0.349	0.302	0.31	*0.59*	0.319	0.297	0.393	0.224	0.349

CT 100	Character trigrams, upper case, no space, stemmed, and after extracting 245 words identified by the TF-IDF threshold 100

Table E.3: Character trigrams after applying TF-IDF words extractions

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
CT	**0.736**	**0.732**	**0.756**	***0.898***	**0.675**	**0.67**	**0.848**	**0.643**	**0.683**
CT 100	0.684	0.612	0.729	*0.848*	0.662	0.62	0.778	0.58	0.657
CT 90	0.658	0.579	0.679	*0.845*	0.65	0.593	0.759	0.522	0.643
CT 80	0.659	0.602	0.693	*0.833*	0.634	0.604	0.768	0.504	0.645
CT 70	0.65	0.569	0.676	*0.832*	0.622	0.591	0.731	0.552	0.63
CT 60	0.626	0.539	0.646	*0.836*	0.562	0.6	0.701	0.53	0.602
CT 50	0.593	0.546	0.591	*0.817*	0.541	0.537	0.688	0.462	0.581
CT 40	0.556	0.496	0.585	*0.766*	0.478	0.519	0.648	0.424	0.545
CT 30	0.498	0.386	0.54	*0.704*	0.442	0.465	0.578	0.374	0.488
CT 25	0.438	0.319	0.479	*0.643*	0.386	0.398	0.477	0.291	0.496
CT 20	0.424	0.314	0.437	*0.64*	0.367	0.392	0.481	0.329	0.418
CT 15	0.352	0.273	0.389	*0.557*	0.279	0.305	0.383	0.254	0.359
CT 10	0.282	0.213	0.229	*0.499*	0.2	0.251	0.399	0.175	0.266

CT 100	Character trigrams, upper case, no space, stemmed, no function words and after extracting 245 words identified by the TF-IDF threshold 100

Table E.4: Character trigrams (no function words) after applying TF-IDF words extractions

Machine Learning Tool: Megam									
Features	Accuracy	Native	Bulgarian	Chinese	Czech	French	Japanese	Russian	Spanish
C4	**0.773**	**0.766**	**0.786**	***0.916***	**0.725**	0.721	**0.866**	**0.675**	**0.739**
C4 100	0.738	0.67	0.781	*0.862*	0.71	**0.752**	0.749	0.636	0.738
C4 90	0.735	0.678	0.777	*0.86*	0.715	0.74	0.753	0.63	0.724
C4 80	0.728	0.655	0.776	*0.856*	0.695	0.729	0.77	0.633	0.697
C4 70	0.713	0.609	0.765	*0.853*	0.679	0.698	0.78	0.622	0.695
C4 60	0.703	0.635	0.752	*0.842*	0.673	0.68	0.777	0.592	0.672
C4 50	0.674	0.592	0.728	*0.807*	0.662	0.64	0.735	0.57	0.653
C4 40	0.644	0.537	0.674	*0.805*	0.604	0.652	0.691	0.535	0.642
C4 30	0.588	0.491	0.653	*0.728*	0.541	0.565	0.683	0.438	0.581
C4 25	0.548	0.456	0.587	*0.707*	0.501	0.507	0.59	0.441	0.565
C4 20	0.507	0.428	0.548	*0.682*	0.449	0.477	0.566	0.372	0.501
C4 15	0.417	0.32	0.49	*0.607*	0.333	0.377	0.492	0.281	0.379
C4 10	0.344	0.246	0.297	*0.534*	0.281	0.33	0.449	0.234	0.306

C4 100	Character quadgrams, upper case, no space, stemmed, and after extracting 245 words identified by the TF-IDF threshold 100

Table E.5: Character quadgrams after applying TF-IDF words extractions

THIS PAGE INTENTIONALLY LEFT BLANK

REFERENCES

[1] Terence Odlin. *Language Transfer: Cross-Linguistic Influence in Language Learning*. Cambridge University Press, 1989.

[2] E. A. Friedman, G. A. Miller, and E. B. Newman. Length-frequency statistics for written english. *Information and Control*, 1:370–389, 1958.

[3] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, 2009.

[4] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: "the penn treebank". *Compuational Linguistics*, 19(2), 1993.

[5] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, second edition, May 2008.

[6] William A. Gale. Good turing smoothing without tears. *Journal of Quantitative Linguistics*, 2, 1995.

[7] Andrew McCallum and Kamal Nigam. *A Comparison of Event Models for Naive Bayes Text Classification*. Number 41-48. AAAI Press, Carnegie Mellon University, Pittsburgh, PA 15213, 1998.

[8] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Compuational Linguistics*, 22(1):39–71, 1996.

[9] Adwait Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD dissertation, University of Pennsylvania, 1998.

[10] Oren Tsur and Ari Rappoport. Using classifier features for studying the effect of native language on the choice of written second language words. *Proceedings of the Workshop on Cognitive Aspects of Computation Language Acquisition*, pp. 9–16, June 2007.

[11] A. Schein (2010). *The Naval Postgraduate School Machine Language Library. Monterey, CA. [Online].* `Available:http://sourceforge.net/projects/npsml/[Accessed:August20,2010]`.

[12] Hal Daume III (2007). *Mega Model Optimization Package [Online].* `Available:http://www.umiacs.umd.edu/~hal/megam/[Acessed:June10,2010]`.

[13] Marie-Catherine de marneffe and Christopher D. Manning. *Stanford Typed Dependencies Manual*. Stanford University, September 2008.

[14] David M. Blei and John D. Lafferty. Topic models. *Text Mining: Theory and Applications.*, 2009.

[15] Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Determining an author's native language by mining a text for errors. Ramat-Gen, 52900, Israel, August 2005.

[16] Sze-Meng Jojo Wong and Mark Dras. Contrastive analysis and native language identification. 2009.

[17] Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. *International Corpus of Learner English: Verison 2*. Universite de Louvain, 2009.

[18] Tysto. (2004). *UK vs US spelling [Online]*, November 2010. `Available:http://www.tysto.com/articles05/q1/20050324uk-us.shtml[Accessed: July2,2010].`

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California